

UNIVERSIDAD NACIONAL TECNOLÓGICA DE LIMA SUR

**FACULTAD DE INGENIERÍA Y GESTIÓN
ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS**



**“EVALUACIÓN DE LA IMPLEMENTACIÓN DE LA PLATAFORMA
IBM API CONNECT V.5.0 PARA LA MEJORA DE LA INTEGRACIÓN
DE LOS WEB SERVICES EN EL BANCO INTERBANK”**

TRABAJO DE SUFICIENCIA PROFESIONAL

Para optar el Título Profesional de

INGENIERO DE SISTEMAS

PRESENTADO POR EL BACHILLER

VICENTE PAUCAR, KARLA STEPHANIE

**Villa El Salvador
2020**

Dedicatoria

A mi madre por el gran apoyo incondicional y fortaleza que me permiten cada día cumplir mis objetivos.

Agradecimiento

Mi eterno agradecimiento a mi familia que gentilmente me apoyaron y orientaron durante el desarrollo de este trabajo de suficiencia y a mi madre que con su apoyo supieron llevarme a luchar y culminar mi carrera.

ÍNDICE

Dedicatoria	ii
Agradecimiento	iii
Resumen	1
Introducción.....	3
Objetivos	5
CAPÍTULO I: MARCO TEÓRICO	6
1.1. Antecedentes:	6
1.1.1. Antecedentes Internacionales.....	6
1.1.2. Antecedentes Nacionales	8
1.2. Bases Teóricas	11
1.2.1. Evaluación	11
1.2.1.1. Métricas de Evaluación.....	11
1.2.1.2. ISO/IEC 25023:2016.....	12
1.2.1.2.1. Características Fundamentales.	12
1.2.1.3. Descripción de las métricas de Evaluación.....	15
1.2.1.3.1. Alcance.	15
1.2.1.3.2. Tiempo de respuesta.	15
1.2.1.3.3. Tiempo de despliegue.....	16
1.2.1.3.4. Calidad.....	16
1.2.1.3.5. Utilización de Recursos.....	17
1.2.2. Implementación de la Herramienta API Connect	17
1.2.2.1. ¿Qué es una API?	17
1.2.2.2. ¿Por qué utilizar API?	18

1.2.2.3.	Descripción API Connect.	19
1.2.2.4.	Ciclo de vida de API Connect.	19
1.2.2.5.	Componentes de API Connect.....	21
1.2.2.5.1.	API Management Node.....	21
1.2.2.5.2.	API Designer.....	23
1.2.2.5.3.	API Gateway.....	25
1.2.2.5.4.	Características de API Gateway.	25
1.2.2.6.	Roles de Usuario de API Connect.	27
1.2.2.7.	Gestión de API en el sector bancario.....	28
1.2.2.8.	Terminologías necesarias para API Connect.....	29
1.2.3.	Web Service.....	35
1.2.3.1.	Definición.	35
1.2.3.2.	Características.....	35
1.2.3.3.	Arquitectura.....	36
1.2.3.4.	Como funciona.....	38
1.2.3.5.	Tipos.	39
1.2.3.5.1.	Tipo SOAP.....	39
1.2.3.5.2.	Tipo REST.	40
1.2.3.5.3.	SOAP VS REST.....	41
1.2.3.6.	Ventajas y Desventajas.....	43
1.2.4.	Metodología de desarrollo de la plataforma.....	43
1.2.4.1.	Definición de Extreme Programming (XP).	43

1.2.4.2.	Actividades de XP.....	44
1.2.4.3.	Prácticas básicas de XP.	46
1.2.4.4.	Ciclos de XP.	46
1.2.4.5.	Actores y Responsabilidades de XP.	47
1.2.4.6.	Artefactos de XP.	50
1.2.5.	Área de integración de Interbank.....	52
1.2.5.1.	Banco Interbank.....	52
1.2.5.2.	Estructura organizacional actual de la empresa.	53
1.2.5.3.	Plataforma de Integración en el banco Interbank.....	54
1.3.	Definición de términos básicos	55
1.3.1.	ALMACENAMIENTO CACHÉ.....	55
1.3.2.	BASE DE DATOS.....	55
1.3.3.	DATA POWER.....	55
1.3.4.	HTTP.....	56
1.3.5.	HTTPS	56
1.3.6.	IBM INTEGRATION BUS.....	56
1.3.7.	URL O URI.....	57
1.3.8.	WINSCP.....	57
1.3.9.	WSDL.....	57
1.3.10.	XML	58
 CAPÍTULO II: METODOLOGÍA DE DESARROLLO DEL TRABAJO		
	PROFESIONAL.....	59

2.1.	Delimitación temporal y espacial del trabajo	59
2.1.1.	Delimitación temporal	59
2.1.2.	Limitación Espacial	59
2.2.	Determinación y análisis del problema	60
2.3.	Modelo de solución propuesto	63
2.3.1.	Recursos para emplear.....	63
2.3.1.1.	Herramienta API MANAGER.	63
2.3.1.2.	Herramienta IBM DataPower Gateway.	64
2.3.1.3.	Herramienta JMETER.....	67
2.3.1.4.	Herramienta Postman.	68
2.3.1.5.	Herramienta SOAPUI.....	69
2.3.1.6.	Herramienta Winscp.	70
2.3.2.	Procedimientos	71
2.3.2.1.	Cronograma de Análisis.....	71
2.3.2.2.	Presupuesto.....	72
2.3.3.	Desarrollo de la implementación de la Plataforma API Connect.....	73
2.3.3.1.	Modelo de la Arquitectura de Integración del Banco Interbank.....	73
2.3.3.2.	Validación API input/output fieldnames.....	74
2.3.3.3.	Contenido de API.....	75
2.3.3.3.1.	Datos.....	75
2.3.3.3.2.	Mensaje.	75
2.3.3.4.	Generaciones de API.....	76

2.3.3.5. HTTP verbs.....	77
2.3.3.6. Códigos de estado de respuesta HTTP.....	79
2.3.4. Desarrollo de la Implementación de la Plataforma API Connect evaluada.....	83
2.3.4.1. Definición de los Requerimientos.....	83
2.3.4.1.1. Servicio Proveedor Watson.....	83
2.3.4.2. Construcción del Servicio SOAP (IBM Integration BUS).....	85
2.3.4.2.1. Arquitectura y Diseño de la construcción del servicio Canónico Generar Token.....	86
2.3.4.2.2. Arquitectura y Diseño de la construcción del servicio Compuesto Enviar SMS.....	87
2.3.4.2.3. Datos Técnicos del servicio Canónico Generar Token.....	88
2.3.4.2.4. Datos Técnicos del servicio Compuesto Enviar SMS.....	89
2.3.4.2.5. Descripción de los campos.....	90
2.3.4.2.6. Ejecución de Servicio SOAP.....	91
2.3.4.3. Construcción del Servicio REST (IBM API CONNECT V.5.0).....	94
2.3.4.3.1. Arquitectura y Diseño de la construcción del servicio REST.....	95
2.3.4.3.2. Datos Técnicos del servicio Canónico Generar Token.....	96
2.3.4.3.3. Descripción de los campos.....	97
2.3.4.3.4. Ejecución de Servicio REST.....	98
2.3.5. Metodología de Evaluación.....	101
2.3.5.1. Métodos, técnicas y métricas de recolección de datos.....	101

2.3.5.1.1. Alcance.....	101
2.3.5.1.2. Calidad.....	102
2.3.5.1.3. Tiempo de respuesta.....	105
2.3.5.1.4. Ejecución de Pruebas de Carga al Servicio API.....	112
2.3.5.1.5. Ejecución de Prueba de Concurrencia al Servicio REST.....	117
2.3.5.1.6. Tiempo de despliegue.....	122
2.3.5.1.7. Utilización de Recursos.....	123
2.3.5.1.8. Tamaño de Espacio de almacenamiento.....	124
2.4. Resultados.....	126
2.4.1. Interpretación del análisis.....	126
2.4.1.1. Líneas de Código.....	126
2.4.1.2. Tiempos de Programación.....	127
2.4.1.3. Tamaño de Espacio de almacenamiento.....	128
2.4.2. Discusión de los Resultados.....	129
CONCLUSIONES.....	131
RECOMENDACIONES.....	132
BIBLIOGRAFÍA.....	133
ANEXOS.....	139
ANEXO 1.....	139
ANEXO 2.....	140
ANEXO 3.....	141

ÍNDICE DE FIGURAS

Figura 1. Representación de la gestión de API's para comunicación entre los sistemas de Backend y diversas plataformas (REDHAT, 2020)	18
Figura 2. Ciclo de vida de IBM API Connect (IBM, 2017).....	21
Figura 3. Portal del API Designer (IBM, s.f.).....	24
Figura 4. Portal del API Gateway (IBM, 2020).....	27
Figura 5. Relación entre Productos, planes y APIs (IBM, 2017).....	30
Figura 6. Jerarquía de Productos Planes y API (IBM, 2017).	32
Figura 7. Organizaciones de proveedores, las organizaciones de desarrolladores y los usuarios (IBM, 2017).....	34
Figura 8. Diagrama de una arquitectura de Web Service orientada a servicios.	38
Figura 9. Concepto básico de servicio SOAP (Sandy, 2014)	40
Figura 10. Servicios orientados a la arquitectura SOA (Blancarte, 2017).....	42
Figura 11. Conjunto de Prácticas de la metodología XP (Universidad de Buenos Aires, 2016).....	46
Figura 12. Fases del ciclo de vida de la metodología XP (Hurtado y Bastiarrica, 2016)	47
Figura 13. Los roles y responsabilidades de XP (Solcataleya, 2016).....	50
Figura 14. Modelo propuesto para una historia de usuario.....	51
Figura 15. Modelo propuesto para una tarea de ingeniería.....	51
Figura 16. Modelo de tarjeta CRC (Universidad de Buenos Aires, 2016).....	52

Figura 17. Organigrama de Interbank (Murguía, 2016).	54
Figura 18. Sede Central del Banco Internacional del Perú INTERBANK (Oficina de consultoría en el sector de la ingeniería civil en lima Perú, 2000).....	60
Figura 19. Interfaz de la herramienta API Manager.....	63
Figura 20. Interfaz de la herramienta Data Power Gateway.	65
Figura 21. Interfaz de la herramienta Data Power operations Dashboard.	66
Figura 22. Interfaz de la herramienta Jmeter.....	67
Figura 23. Interfaz de la herramienta Postman.....	68
Figura 24. Interfaz de la herramienta SOAPUI.	69
Figura 25. Interfaz de la herramienta WINSOAP.....	70
Figura 26. Arquitectura de Integración del Banco Interbank.....	73
Figura 27. Validación de Integración del Banco Interbank.....	74
Figura 28. Evolución de las Generaciones de API.....	76
Figura 29. Estructura GET.....	77
Figura 30. Estructura POST.	77
Figura 31. Estructura PUT.....	78
Figura 32. Estructura DELETE.	78
Figura 33. Códigos de respuesta.....	79
Figura 34. Estructura Response exitoso.....	79
Figura 35. Estructura Response error.....	80
Figura 36. Estructura Response no autorizada.	80

Figura 37. Estructura Response error de negocio.	81
Figura 38. Estructura Response recurso no encontrado.	81
Figura 39. Estructura Response error exceso de solicitudes.....	82
Figura 40. Estructura Response error de servidor.	82
Figura 41. Arquitectura de servicio SOAP.	85
Figura 42. Diseño del servicio SOAP – Canónico.....	86
Figura 43. Diseño del servicio SOAP – Compuesto.	87
Figura 44. Request del servicio SOAP.	92
Figura 45. Response del servicio SOAP.....	93
Figura 46. Arquitectura de servicio REST.....	94
Figura 47. Diseño del servicio API.....	95
Figura 48. JSON Request para consumo del servicio API	98
Figura 49. Request del servicio REST.....	99
Figura 50. Response del servicio REST.	100
Figura 51. Archivo de configuración del script de Carga.	105
Figura 52. Parametrización de los campos de entrada.....	106
Figura 53. Configuración del Data SET.	107
Figura 54. Archivo Datapool.csv.	107
Figura 55. Configuración del grupo de hilos para la ejecución de carga.	108
Figura 56. Lista de las ejecuciones realizadas.	108
Figura 57. Lista de los tiempos de muestra de las ejecuciones realizadas. ..	109
Figura 58. Reporte de resumen de la prueba de carga ejecutada.....	109

Figura 59. Archivo de configuración del script de Concurrencia.....	110
Figura 60. Lista de las ejecuciones realizadas.	110
Figura 61. Configuración del grupo de concurrencia para la ejecución de carga.....	111
Figura 62. Lista de los tiempos de muestra de las ejecuciones realizadas. ..	111
Figura 63. Reporte de resumen de la prueba de concurrencia ejecutada.	112
Figura 64. Archivo de configuración del script de Carga.	113
Figura 65. Parametrización de los campos de entrada.....	114
Figura 66. Configuración del Data SET.	114
Figura 67. Archivo Datapool.csv.	115
Figura 68. Configuración del grupo de hilos para la ejecución de carga.	115
Figura 69. Lista de las ejecuciones realizadas..	116
Figura 70. Lista de los tiempos de muestra de las ejecuciones realizadas. ..	116
Figura 71. Reporte de resumen de la prueba de carga ejecutada.....	117
Figura 72. Archivo de configuración del script de Concurrencia.....	118
Figura 73. Lista de las ejecuciones realizadas.	118
Figura 74. Configuración del grupo de concurrencia para la ejecución de carga.....	119
Figura 75. Lista de los tiempos de muestra de las ejecuciones realizadas.. .	119
Figura 76. Reporte de resumen de la prueba de concurrencia ejecutada.	120
Figura 77 Estructura de Historia de Cambios Base	139
Figura 78 Plantilla para Solicitar Uso de Servicios del API.....	140

ÍNDICE DE TABLAS

Tabla 1 Comparación de Soap vs Rest.	42
Tabla 2 Cronograma de actividades	71
Tabla 3 Presupuesto de la Herramienta	72
Tabla 4 Datos generales del servicio Canónico.....	88
Tabla 5 Datos generales del Servicio Compuesto.....	89
Tabla 6 Detalle de campos.	90
Tabla 7 Detalles técnicos de BUS	93
Tabla 8 Detalle de datos técnicos.....	96
Tabla 9 Detalle de datos de los campos.....	97
Tabla 10 Detalles técnicos de API.....	100
Tabla 11 Evaluación de la Métrica de Alcance.	101
Tabla 12 Valor de medición de la categoría Confiabilidad.....	102
Tabla 13 Valor de medición de la categoría Funcionalidad.	102
Tabla 14 Valor de medición de la categoría Usabilidad.....	103
Tabla 15 Evaluación de la métrica de calidad – Plataforma Integration BUS.....	103
Tabla 16 Evaluación de la métrica de calidad – Plataforma Integration API CONNECT.	104
Tabla 17 Evaluación de la métrica de calidad – Plataforma Integration BUS vs API Connect.	104

Tabla 18 Resumen de la prueba de carga al servicio BUS	109
Tabla 19 Resumen de la prueba de concurrencia al servicio BUS.....	112
Tabla 20 Resumen de la prueba de carga al servicio API.....	117
Tabla 21 Resumen de la prueba de concurrencia al servicio API.	120
Tabla 22 Evaluación de los tiempos de respuesta según la plataforma.	121
Tabla 23 Evaluación del tiempo de despliegue.	122
Tabla 24 Evaluación de la cantidad de líneas de Código.	123
Tabla 25 Evaluación de la cantidad de espacio de almacenamiento.....	124
Tabla 26 Cantidad de Líneas de Código	126
Tabla 27 Tiempos de Programación.....	127
Tabla 28 Tamaño de Espacio de almacenamiento.....	128

Resumen

Los Webservice se definen como un sistema de software designado para dar soporte a la interacción de máquina a máquina a través de una red, estos realizan una tarea o un conjunto de tareas específicas, y se describe mediante una descripción de servicio en una notación estándar XML llamada WSDL (Web Services Description Language), la cual proporciona todos los detalles necesarios para interactuar con el servicio, incluidos los formatos de mensaje que detallan tanto las operaciones como los protocolos de transporte y la ubicación.

Interbank para la realización de sus diversas operaciones bancarias como altas de un cliente, altas de tarjetas, apertura de cuentas, transferencias de dinero, consultas de saldos, etc. En su plataforma de integración utiliza los Webservices como solución de intercambio de datos entre sus aplicaciones distribuidas que se ejecutan en diferentes sistemas operativos y arquitecturas.

Para llevar a cabo el estudio se ha estructurado en 03 capítulos. En el capítulo I "Marco Teórico" se presentan las precisiones teórico conceptuales de los Web services, la plataforma IBM API Connect y otros conceptos que les permitan comprender el tema de estudio. En el capítulo II "Metodología de desarrollo del trabajo profesional" se describe la delimitación temporal y espacial donde he realizado el trabajo, se presenta la situación actual de la plataforma de integración del banco. Asimismo, se desarrolla la propuesta para el desarrollo de

los Web services bajo la plataforma IBM API Connect y se aplican las métricas que permitieron comparar las plataformas de integración actual y la plataforma IBM API Connect.

Finalmente, en el Capítulo III "Conclusiones" se determina si la implementación de la plataforma propuesta permite mejorar la integración en el banco Interbank.

Introducción

Actualmente las aplicaciones comerciales utilizan diversas plataformas de programación que permiten agrupar los servicios con los que cuenta la empresa, las empresas alrededor del mundo buscan exponer sus servicios a más clientes a través de internet, por lo que hacen uso de los servicios web o web services para asegurar la comunicación entre cliente - servidor en la World Wide Web.

En los últimos años las compañías de los diferentes sectores de nuestro país, especialmente las compañías del sector financiero aumentaron su competitividad, estas se encuentran en medio de un proceso de transformación digital que les permita ampliar sus canales de comunicación con sus clientes, lo que implica contar con una plataforma de integración de servicios web que se ejecuten en diversos sistemas operativos y arquitecturas; y que brinde la confiabilidad, seguridad y rapidez necesaria para la utilización de sus servicios.

El Banco Internacional del Perú S.A.A. (Interbank Perú) es una empresa que opera principalmente en el sector de Seguros y Banca; para la realización de sus diversas operaciones bancarias como altas de un cliente, altas de tarjetas, apertura de cuentas, transferencias de dinero, consultas de saldos, etc., en su plataforma de integración basados en el protocolo SOAP utiliza los Web services como solución al intercambio de datos entre sus aplicaciones divididas y distribuidas.

Para que el banco Interbank pueda mantenerse a la vanguardia de la tecnología es necesaria la implementación de una plataforma de integración API en los desarrollos de los Web services que le permita mejorar el performance en su comunicación de datos, organizar su comunicación interna para dar apoyo a nuevos proyectos innovadores de manera uniforme, reducir costos de mantenimiento y despliegue de servicios e incrementar la agilidad de los procesos de transformación.

Objetivos

a) General:

- Evaluar la implementación de la plataforma IBM API Connect v.5.0 para la mejora de la integración de los web services en el banco Interbank.

b) Específicos:

- Planificar y determinar las métricas que se utilizarán para la medición de las plataformas.
- Evaluar los resultados de la medición de las métricas empleadas.
- Demostrar que la plataforma IBM API CONNECT V.5.0. es la más adecuada para la integración de los Web Services.

CAPÍTULO I: MARCO TEÓRICO

1.1. Antecedentes:

1.1.1. Antecedentes Internacionales

A) Puerta (2015) realizó la investigación: “**Desarrollo de una API para la descripción y gestión de Servicios Web REST**”, cuyo objetivo principal es diseñar e implementar una API REST para la gestión de descripciones de servicios, la investigación llegó a las siguientes principales conclusiones:

- Se diseñaron y desarrollaron dos API, una para la descripción de servicios y otra para su monitorización y gestión.
- Se pidió un listado de todas las descripciones almacenadas en el servidor y se ha comprobado que se recupera correctamente.
- Se envió una descripción de monitorización con un nombre de servicio que existe en el repositorio de descripciones de servicios y se ha comprobado que se crea correctamente.

La implementación de las API se realizó siguiendo el estilo arquitectónico REST que resultó ser más ligero y fácil de implementar que SOAP.

La importancia de este antecedente en la aplicación del trabajo presente, es comprobar que la descripción de servicios es más ligera en REST que

en SOAP, puesto que en SOAP obliga a utilizar WSDL; en cambio, REST puede utilizar cualquier tipo de descripción que en el caso de la tesis utilizaron formato JSON.

B) Carmona (2014), realizó la investigación: **“Plataformas de Integración: Servicios web basados en las arquitecturas SOAP y REST”**, cuyo objetivo principal es comparar los aspectos y descubrir los factores que impactan en el rendimiento de los servicios web SOAP y REST, sostiene que SOAP ocupa más trabajo, por lo que la investigación da a conocer que:

- Inclinarsen de modo conveniente por los Web services utilizando la arquitectura REST para mostrar gran cantidad de datos como es el caso de algunas grandes empresas.
- REST es una arquitectura que permite crear aplicaciones “cliente-servidor”, mientras que SOAP funciona como un protocolo que permite intercambiar todos los datos que se encuentren entre 2 extremos.

Finalmente, Las afirmaciones concluyen que los valores de la arquitectura REST es mucho menor que los atributos del servicio web basado en SOAP y sugieren que SOAP debe ser usado cuando se establece un documento formal que necesite el detalle las interacciones que tendrá la interfaz y REST si nos encontramos cuando el web service no necesita tener estado.

C) Rojas y Tafur (2014) realizaron la investigación: **“Marco de referencia para la orquestación de servicios sobre ESB Open Source suministrando un conjunto de buenas prácticas que sirvan como instrumento a un arquitecto de software para la implementación de una SOA”**, cuyo principal objetivo es proponer un conjunto de buenas prácticas para orquestación de servicios para la implementación de una SOA y las principales observaciones fueron:

- EL Lenguaje de Ejecución de Procesos (BPEL) tienen un fácil manejo luego de estudiar al API de cada EBS (Bus de Servicio Empresarial).
- Que los Lenguajes de Ejecución de Procesos (BPEL), desarrollados en java son más flexibles y fáciles de depurar para el momento de realización de pruebas.

La importancia de este antecedente en la aplicación del trabajo presente, es revisar los diferentes ESB que existen para la implementación de arquitectura web services SOA.

1.1.2. Antecedentes Nacionales

A) Medina y Pinedo (2010) realizaron la investigación: **“Implementación de un sistema de integración para las bibliotecas municipales de Lima y Callao utilizando SOA y J2ME tesina para optar el título profesional de ingeniero de**

sistemas”, la investigación llegó a las siguientes principales conclusiones:

- La implementación de una Arquitectura Orientada a Servicios, la utilización de Web Services para definir cada servicio, la implantación de un Bus de Servicios Empresarial para llevar a cabo la integración permitirá definitivamente la integración de información de las bibliotecas municipales de Lima y Callao.
- La implementación de esta solución se brindará un mejor servicio a los usuarios de las bibliotecas municipales facilitándoles el acceso a la información del material bibliográfico, contribuyendo en algo a la investigación y educación de la población.

La importancia de este antecedente en la aplicación del trabajo presente, es comprender como los servicios SOA se reutilizan con facilidad y se ensamblan enseguida en nuevas aplicaciones compuestas.

Además, el espacio de estudio, se localiza en nuestra ciudad, permitiendo tener una visión más clara de la problemática presente en la región con respecto al uso de arquitectura Orientada a Servicios (Web services).

B) Mamani y Atencio (2017) realizaron la investigación: **“Implementación basado en API REST para la mejora de la interconectividad en las aplicaciones web de la Municipalidad**

provincial de Lampa”, cuyo objetivo principal es Desarrollar un Modelo de Comunicación Basado en API REST, la investigación llegó a la siguiente principal conclusión:

- Se logró implementar y desarrollar un prototipo de comunicación basado en API REST en la mejora de las aplicaciones de la Municipalidad Provincial de Lampa mediante entrevistas al personal que trabajan en las áreas administrativas evidenciando la mejora en los tiempos y procesos.

La importancia de este antecedente en la aplicación del trabajo presente, es que se comprobó que existe una mejora significativa al usar el modelo de comunicación API REST, y que se facilita la lectura de datos con el uso estándar JSON.

C) Burgos (2017) realizó la investigación: “Análisis y evaluación de las arquitecturas REST y SOAP para el desarrollo de servicios de web aplicados al ERP "ADRISERP" y su versión móvil en Android”, cuyo principal objetivo es evaluar las arquitecturas REST y SOAP, para el desarrollo web services, se llegó a las siguientes conclusiones:

- De la interpretación y correcta discusión de todos los resultados del rendimiento de los servicios web, se evidenció que el mejor resultado fue REST.
- Del análisis de los indicadores se determinó que la mejor arquitectura para desarrollar servicios Web es con REST.

La importancia de este antecedente en la aplicación del trabajo presente es que se demuestra que en la comparación de REST y SOAP, Rest es más óptimo tanto en rendimiento y escalabilidad, a su vez permite una curva de aprendizaje y tiempo de desarrollo más rápidos.

1.2. Bases Teóricas

1.2.1. Evaluación

Denominamos a la evaluación como la acción y efecto de evaluar. (Anónimo, 2018) La palabra deriva de evaluar, que proviene del francés évaluer, que tiene por significado determinar el valor de algo'.

Por consiguiente, una evaluación es un juicio cuya finalidad es establecer, tomando en consideración un conjunto de criterios o normas, el valor, la importancia o el significado de algo.

1.2.1.1. Métricas de Evaluación.

Las métricas que se consideraron necesarias para la evaluación del presente trabajo de suficiencia son: alcance, tiempo, calidad y *utilización de Recursos*.

1.2.1.2. ISO/IEC 25023:2016.

Para el presente trabajo se tomará como guía de evaluación al ISO/IEC 25023:2016 - Evaluación de la calidad del producto software

Es un estándar internacional creado para evaluar la calidad de software, esta norma una revisión de la ISO/IEC 9126:2004(Evaluación de la calidad del software). Abud (2012) indica que esta norma se estableció para la evaluación de la calidad de productos de Software y sugiere métricas para la medición de artefactos teniendo en cuenta diferentes criterios asociados al software.

1.2.1.2.1. Características Fundamentales.

Funcionalidad. Es la capacidad del producto de software para proporcionar funciones que satisfacen las necesidades declaradas e implícitas cuando el software es utilizado en condiciones específicas. En este grupo se establecen los siguientes atributos:

- *Adecuación*, evalúa si el software cumple las tareas específicas definidas con un conjunto adecuado de funciones.
- *Exactitud*, evalúa si el software cumple con las necesidades para el que fue creado.
- *Interoperabilidad*, permite evaluar la capacidad del software para interactuar con uno o más sistemas especificados.
- *Seguridad*, mide la capacidad del software para la protección de la información, previniendo los accesos no autorizados.

- *Conformidad*, evalúa si el software tiene la capacidad de cumplir con los estándares relacionados con la funcionalidad.

Confiabilidad. Es la capacidad del producto de software para mantener un nivel de rendimiento estable cuando se utiliza en condiciones específicas, *como subcategoría sugiere:*

- *Recuperación*, relacionado a la capacidad de reestablecer el nivel de Operación, tanto en la recuperación de datos como el esfuerzo que implica lograrlo.

Usabilidad. Es un conjunto de atributos que miden la capacidad del producto de software para ser entendido, aprendido y utilizado por el usuario, en esta característica se agrupan:

- Comprensibilidad que tienen los usuarios para reconocer la estructura lógica.
- Facilidad de aprender el uso, la operación y control del software.

Eficiencia. Se mide la capacidad del producto de software para proporcionar rendimiento en relación con la cantidad de recursos utilizados, en este grupo se presentan los factores:

- Comportamiento con respecto al tiempo tanto el de respuesta como el de procesamiento de los datos.
- Comportamiento con respecto a los recursos.

Mantenibilidad. Permite medir el esfuerzo para modificar el producto de software, estas modificaciones pueden incluir corrección de errores, mejoras o adaptación del software a cambios del ambiente, en este caso se tienen los siguientes atributos:

- Capacidad de análisis, mide el esfuerzo para diagnosticar fallas o partes que necesitan ser modificadas.
- Capacidad de modificación.
- Estabilidad, mide los riesgos de efectos en el software debido a modificaciones.
- Facilidad de prueba, relacionado al esfuerzo que se requiere para la validación del software tras una modificación.

Portabilidad. En este grupo se mide la capacidad del producto de software para ser transferido desde un ambiente a otro, las subcategorías que el estándar sugiere son:

- Adaptabilidad que tiene el software para trabajar en diferentes ambientes.
- Facilidad de despliegue, mide el esfuerzo necesario para el despliegue en un ambiente específico.
- Capacidad de reemplazo, relacionado al esfuerzo utilizado para sustituirlo por otro producto con funciones similares.

1.2.1.3. Descripción de las métricas de Evaluación.

Según el Instituto de Ingenieros Eléctricos y Electrónicos (IEEE), define como métrica como “una medida cuantitativa del grado en que un sistema, componente o proceso posee un atributo determinado”.

Para el desarrollo de este trabajo se han considerado la utilización de métricas basadas en la ISO 25023 las cuales se describen a continuación:

1.2.1.3.1. Alcance.

Relacionada con la subcategoría de Interoperabilidad, según la ISO/IEC 25023:2016, esta permite evaluar la capacidad del producto de software para interactuar con uno o más sistemas especificados, en este trabajo se medirá la capacidad que tiene la plataforma de integración para conectarse con los canales consumidores (Apps, webs, Cloud, ATM, etc.).

1.2.1.3.2. Tiempo de respuesta.

Se realizará basada en la subcategoría de comportamiento temporal, según la ISO ISO/IEC 25023:2016 pertenece al grupo de eficiencia, la cual mide la capacidad del producto de software para proporcionar rendimiento, en relación con la cantidad de recursos utilizados, en el presente trabajo esta métrica se utilizará teniendo en cuenta el tiempo que demora en responder las peticiones enviadas desde el servicio hacia el backend.

1.2.1.3.3. *Tiempo de despliegue.*

En esta característica se mide la capacidad del producto de software para ser transferido desde un ambiente a otro, la ISO 25023 sugiere el atributo de Facilidad de despliegue que mide el esfuerzo necesario para el despliegue en un ambiente específico, esta métrica no servirá para poder verificar el tiempo y complejidad que les toma a los operadores ejecutar un despliegue de un servicio en los ambientes correspondientes.

1.2.1.3.4. *Calidad.*

Según las Normas ISO, se trata de un conjunto de características inherentes que cumple con los requisitos; Abud (2012) señala que cualquier componente de la calidad de software puede ser descrito en términos de una o más de las seis características básicas que sugiere la ISO 9126; para la presente investigación se medirá en función a tres de los atributos de la ISO tales como funcionalidad, descrita como la capacidad de proporcionar funciones que satisfacen las necesidades declaradas e implícitas del software; confiabilidad que se relaciona con la capacidad de mantener la estabilidad en el nivel de rendimiento y la categoría usabilidad que es el conjunto de atributos que miden la capacidad del software para ser entendido, aprendido y utilizado, estas categorías se medirán desde el punto de vista de los expertos del banco que interactúan con la plataforma de integración.

1.2.1.3.5. Utilización de Recursos.

Esta métrica permitirá evaluar la Eficiencia en el desempeño relacionada con la cantidad de líneas de código, utilización de memoria y almacenamiento.

1.2.2. Implementación de la Herramienta API Connect

1.2.2.1. ¿Qué es una API?

IBM(s.f.) describió una API como un conjunto de reglas y especificaciones (en forma de código) que las aplicaciones pueden seguir para lograr la comunicación entre ellas, relaciona al API como un equivalente a la interfaz que los softwares utilizan para comunicarse con los humanos, las API son un interfaz de comunicación entre distintos softwares. Asimismo, refiere que las API pueden ser muy valiosas ya que permiten hacer uso de capacidades, infraestructura y funciones ya presentes en otro software para optimizar sus recursos, rentabilizar sus datos y evitar el retrabajo.

De manera similar, REDHAT (2020) expresó que una API es un conjunto de definiciones y protocolos que se utiliza para desarrollar e integrar el software de las aplicaciones, estas permiten que exista comunicación entre productos y servicios con otros, sin tener la necesidad de saber cómo estos están implementados lo que simplifica el desarrollo de las aplicaciones permitiendo ahorrar tiempo y dinero.

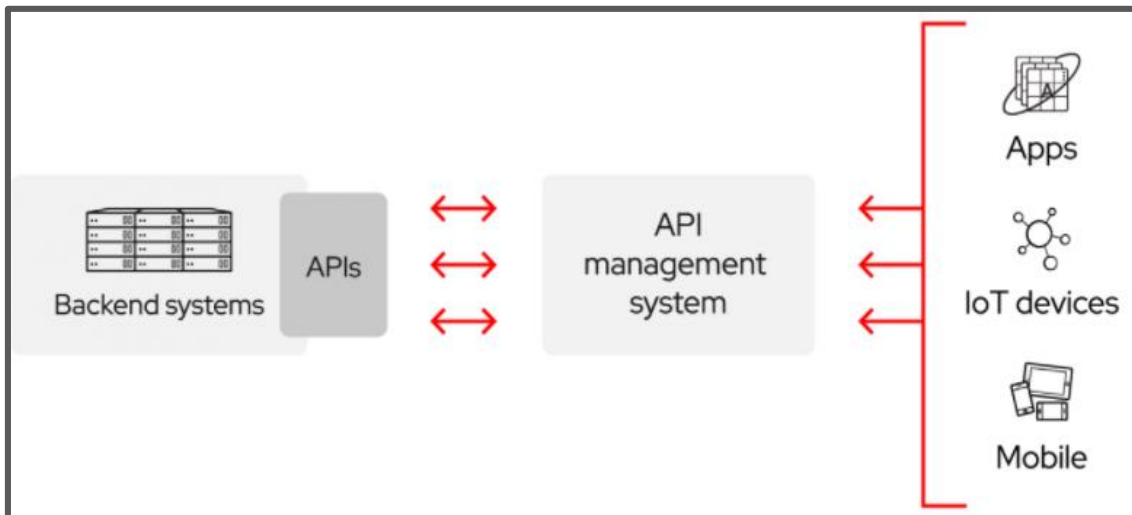


Figura 1. Representación de la gestión de API's para comunicación entre los sistemas de Backend y diversas plataformas (REDHAT, 2020)

1.2.2.2. ¿Por qué utilizar API?

Según IBM (2020) se puede utilizar una API para dar a conocer la empresa ya que los activos, datos o servicios de su empresa se pueden proporcionar a desarrolladores de aplicaciones externos para expandir la empresa y abrir nuevos mercados.

Por su parte BBVAOPEN4U (2016) manifestó que, en medio de un proceso de transformación digital en prácticamente cualquier sector empresarial, el uso de una API resulta esencial en las mejoras de los procesos de un negocio, revelando que existe 04 ventajas que aportan las API en la empresa:

- Reestructuran y organizan sistemas internos para dar apoyo a nuevos proyectos innovadores de una manera uniforme.
- Reducen los costes de mantenimiento.
- Incrementan la agilidad de los procesos de transformación.

- Aportan nuevas oportunidades para generar nuevas formas de llegar a los clientes, generar ingresos y construir nuevas alianzas con socios.

1.2.2.3. Descripción API Connect.

API Connect es una plataforma de API creada por IBM, con la que brinda soluciones para crear, exponer y gestionar las API'S en las organizaciones, construida sobre una arquitectura de microservicios permitiendo que su desarrollo sea en un entorno multi-cloud.

IBM API Connect proporciona un control optimizado en todo el ciclo de vida de la API y también permite a las empresas obtener información detallada sobre el consumo de API a partir de sus análisis integrados.

Gonzales (2019), indicó que API Connect "Es una plataforma que permite conectar clientes finales a un servidor de una empresa para consumir ciertos productos. Esta plataforma es un paso intermedio que permite controlar qué cliente se puede conectar y a qué recursos. Es decir, da una capa de seguridad y de filtrado, haciendo la conexión al servidor de dicha empresa más fiable".

1.2.2.4. Ciclo de vida de API Connect.

Según IBM (2017), los pasos del ciclo de vida de un API son cuatro, incluyen la creación, la ejecución, la gestión y la protección de las API y los define de la siguiente manera:

Crear. En este paso se desarrolla y escribe la definición y la implementación de API. Asimismo, se puede realizar la prueba inicial de la API.

Ejecutar. Aquí se realiza el empaquetado y despliegue del API. Se debe tener en consideración de que la API se encuentre alojada de forma segura en una plataforma estable.

Gestionar. Lo que se realiza en este ciclo es la creación y gestión de portales de autoservicio que expongan el API a los consumidores de API. Además, se puede supervisar el conjunto de reglas y condiciones que gobiernan el API para garantizar que cumple su finalidad, y realizar los ajustes necesarios. En ese mismo contexto permite retirar y archivar el API cuando proceda.

Proteger. En este ciclo se incorpora el control de accesos, la supervisión y el registro cronológico para proteger adecuadamente el API.

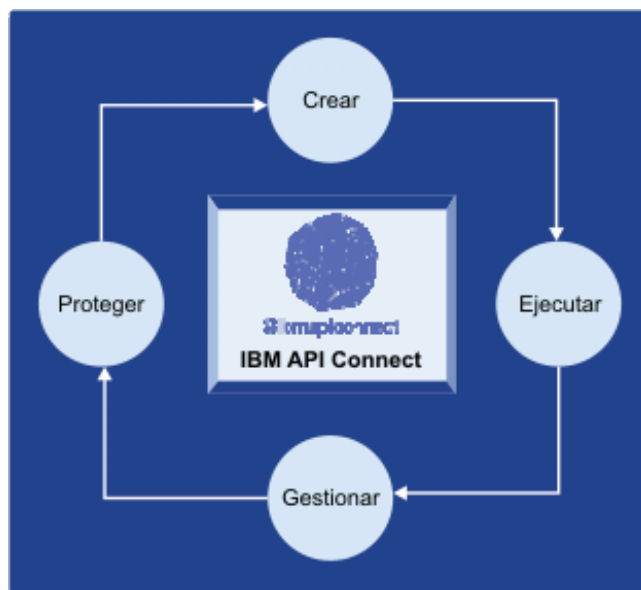


Figura 2. Ciclo de vida de IBM API Connect (IBM, 2017)

1.2.2.5. Componentes de API Connect.

1.2.2.5.1. API Management Node.

Con él se puede controlar el acceso a la plataforma de usuario final puede ser el externo, partner o interno, dividiendo los roles de usuarios y desarrolladores.

Para Glickenhause (2019) la plataforma de administración de API proporciona un conjunto completo de capacidades para cubrir todo el ciclo de vida de una API, desde su creación hasta su implementación y administración. Debe ser una base integrada de creación, tiempo de ejecución, administración y seguridad para las API de nivel empresarial para exponer los activos comerciales centrales y los microservicios para impulsar las aplicaciones digitales modernas.

Las capacidades clave para una plataforma de gestión de API incluyen:

- *Opciones automatizadas, visuales y de codificación para crear API.*
Un conjunto de herramientas para diseñar, modelar, desarrollar, probar e implementar rápidamente API en un modelo de entrega continua automatizado.
- *Ciclo de vida y gobernanza de las API, productos y planes.* La producción de las API, su empaquetado y catalogación y el seguimiento de su ciclo de vida son actividades que ayudan a la gestión y el control efectivos de las API a medida que se implementan.
- *Control de acceso sobre API, planes de API y productos de API.*
Otra función clave para la seguridad es administrar el acceso a API en varios niveles de granularidad que involucran a usuarios y grupos de usuarios en un rol de consumidor o proveedor.
- *Portal de desarrollador de autoservicio personalizable para publicar y consumir API.* Publicar y socializar API a través de un portal fácil de usar es crucial para promover el acceso a su negocio principal y al alcance de mercado de su marca.
- *Soporte Polyglot para la creación de microservicios.* El soporte Polyglot es clave para permitir la innovación y la agilidad dentro de los diferentes modelos de programación requeridos por diferentes escenarios de casos de uso. El soporte para tiempos de ejecución de Node.js y Java, entre otros, es esencial.
- *Compatibilidad con políglotas para ejecutar microservicios.* La ejecución en tiempo de ejecución está respaldada por

características de la plataforma, como el rendimiento, la escalabilidad, el equilibrio de carga y la conmutación por error.

- *Aplicación de políticas, seguridad y control.* Una puerta de enlace de seguridad API escalable y de alto rendimiento es imperativa en cualquier plataforma de administración de API, principalmente para proteger el acceso a sus sistemas de backend.
- *Análisis avanzado de uso de API.* El análisis de las métricas de uso de API desde diferentes perspectivas y roles de usuario ayuda a proporcionar un circuito de retroalimentación a los propietarios y desarrolladores de API para futuras mejoras.
- *Soporte de nube híbrida, multi-nube y en las instalaciones.* *Flexibilidad* para respaldar los activos donde sea que estén y para implementar los componentes de administración de API donde lo desee.
- *Escalabilidad.* La plataforma debería permitirle empezar con poco y crecer tanto como necesite. Esto incluye el tráfico de API, los desarrolladores de aplicaciones (consumidores) y los propietarios y desarrolladores de API; cada uno escala de forma independiente para manejar los picos según sea necesario en esa área sin tener que escalar toda la plataforma.

1.2.2.5.2. API Designer.

Permite hacer pruebas locales, este componente permite diseñar e implementar un mini management y un mini Gateway local.

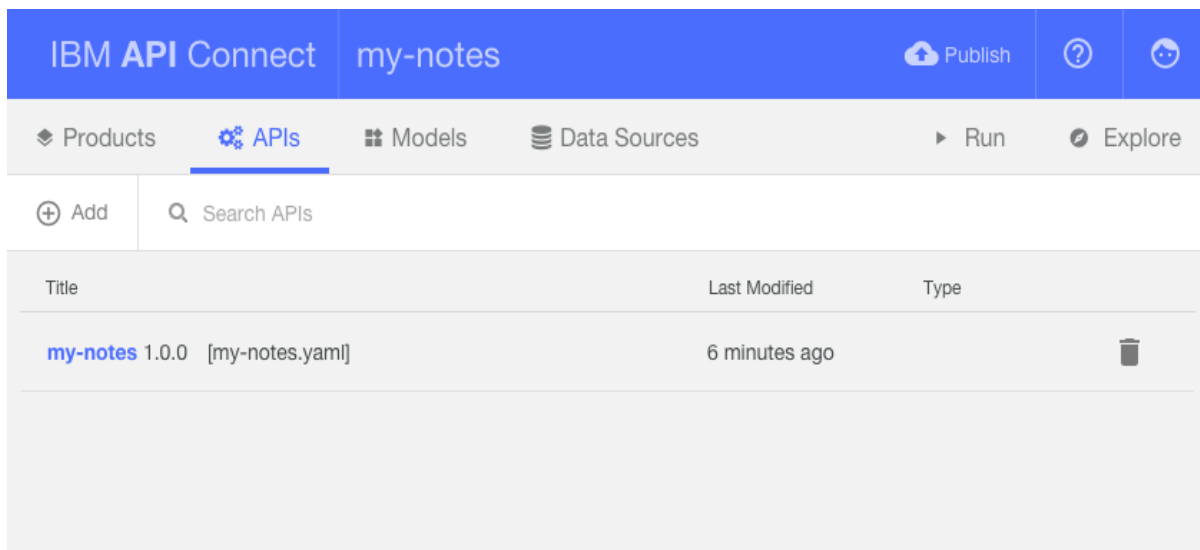


Figura 3. Portal del API Designer (IBM, s.f.)

Kocen (2017) describe que API Designer permite controlar todos los aspectos de API sin tener que realizar un seguimiento de todo en la línea de comandos. En API Designer se puede configurar:

- Los metadatos de la API, incluidos los términos de uso, la información de contacto y los enlaces a la documentación externa.
- Construcción básica de la API, incluidos los esquemas, la ruta base y si la API acepta y genera aplicaciones JSON, XML o un tipo de medio adicional.
- Definiciones de seguridad para administrar la autenticación y autorización de API. Actualmente, API Connect es compatible con la autenticación básica de URL / registro de usuarios, API Key u OAuth.
- Definiciones de seguridad a API específicas. API Connect admite el uso de múltiples opciones de seguridad en una sola API.

- Definiciones su esquema de API para estandarizar las llamadas a API de los desarrolladores.
- Definiciones de las propiedades, rutas y parámetros de su API.

1.2.2.5.3. API Gateway.

Permite usar conexiones a base de datos como Mongo o app externas como API Toolkit con el fin de reforzar el acceso a microservicios y APIS.

IBM (2020) señala que API Gateway es un servicio que puede utilizar para crear, proteger, compartir y gestionar API que acceden a los recursos de IBM Cloud.

API Gateway funciona interceptando la llamada API entrante y ejecuta políticas de seguridad, luego enruta la llamada a la aplicación back-end. Cuando se termina de procesar la solicitud; la aplicación de back-end envía la respuesta a la puerta de enlace, que luego la enruta al usuario que realizó la llamada.

1.2.2.5.4. Características de API Gateway.

Según IBM (2020), API Gateway ofrece las siguientes características para controlar el uso, aumentar la adopción y realizar el seguimiento de las estadísticas de las API.

- **Limitación de velocidad.** Se puede especificar un límite de velocidad de modo que solo se pueda hacer un número permitido de llamadas por segundo, minuto u hora.

- **OAuth.** El protocolo de autorización basado en señales OAuth permite a sitios web o aplicaciones de terceros acceder a datos de usuario sin que el usuario tenga que compartir información personal.
- **CORS (Cross-Origin Resource Sharing).** Por sus siglas en español significa Compartición de recursos entre orígenes, es un mecanismo que utiliza cabeceras HTTP para permitir que una app recupere los datos de una API más allá de los límites del dominio.
- **Análisis de API.** Se puede almacenar y visualizar la información generada durante las invocaciones de API, se puede supervisar el uso para entender cómo se utilizan las API de modo que permite tomar decisiones, alguna de las estadísticas que se pueden obtener de las API son:
 - El número de respuestas y el tiempo medio de respuesta durante la última hora o durante el intervalo de tiempo que especifique.
 - El número de llamadas de API por minuto.
 - Las últimas 100 respuestas.

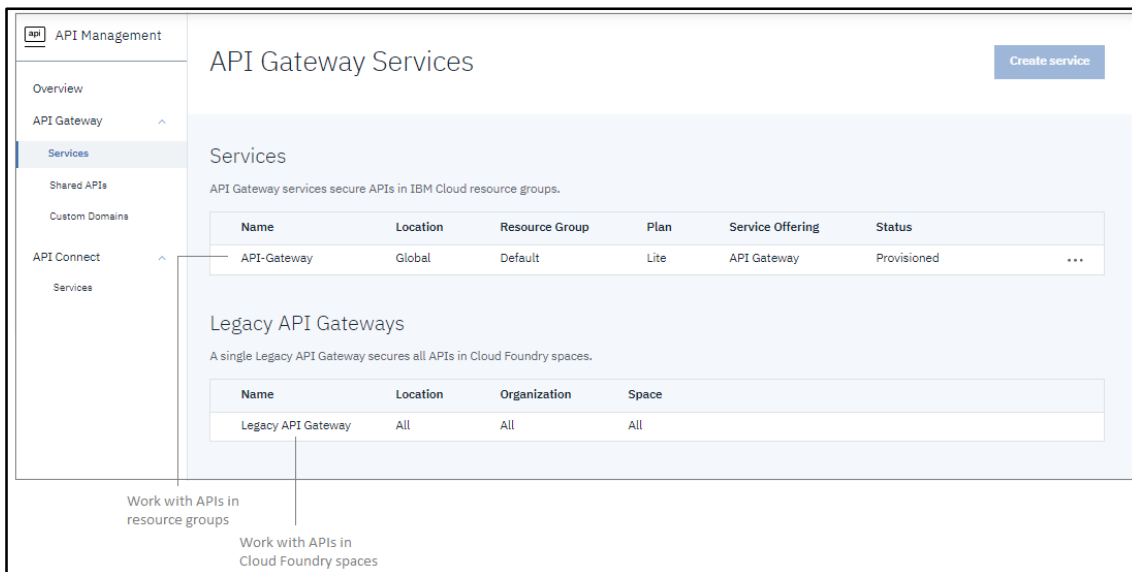


Figura 4. Portal del API Gateway (IBM, 2020)

1.2.2.6. Roles de Usuario de API Connect.

IBM API Connect proporciona una infraestructura, herramientas e instalaciones que permiten a los usuarios crear, gestionar y organizar API. La capacidad para realizar tareas en las interfaces de usuario de IBM API Connect se controla mediante roles de usuario y los permisos que se asignan a esos roles. A continuación, se describen los roles de API Connect predeterminados. IBM (2020)

Owner. Por ser el propietario tiene acceso completo y no se puede editar ni eliminar. Todos los demás roles, incluidos los roles personalizados, se pueden eliminar. Si elimina una función, los usuarios pierden esa función. Si un usuario pierde ese rol, su cuenta permanece en API Manager, lo que le permite agregar un rol al usuario en una fecha futura.

Administrator. Es el responsable de coordinar los recursos de productividad y planeación del proyecto, al igual que el rol Owner, cuenta con todos los permisos y puede realizar todas las acciones disponibles en el API Manager.

Product Manager. Es la persona que debe asegurar la gestión del producto, este rol tiene permisos para gestionar los roles API Developers, puede aprobar suscripciones; además que tiene acceso a poder editar los borradores de Productos y borradores API´s.

API Developer. Encargado de desarrollar e implementar las API's, los desarrolladores tienen permisos de editar los borradores de Productos y borradores API´s.

Publisher. El editor es el que se encarga de publicar el producto final, este tiene permisos para editar la administración de catálogos y aprobar suscripciones.

1.2.2.7. Gestión de API en el sector bancario.

El sector bancario para mantenerse siendo competitivos y ganar nuevos clientes, ha sabido aprovechar las ventajas de las API'S para no solo agilizar sus procesos si no que dotarlos con mayor seguridad, en este sector la aplicación de las API´s va desde mejorar los métodos de pago hasta luchar contra el riesgo y fraude cibernético.

Para Cybergestion (2018), una de las principales ventajas de las API´s para el sector financiero son la automatización de procesos, permitiendo la disminución de costes al reducir el uso de otros canales de comunicación como los centros de llamadas. Por otro lado, las API´s han permitido la personalización de los servicios ofrecidos mejorando notablemente la experiencia del cliente con los servicios del banco, por ejemplo, teniendo contacto más directo con cada usuario a través de

notificaciones a su móvil y correo electrónico, brindándole el autoservicio online asegurando una respuesta más rápida y eficiente a las necesidades de los clientes. Por último, estas interfaces han generado nuevos ingresos a las entidades bancarias, ya que permiten que terceras empresas puedan acceder a datos y servicios propios para mejorar sus negocios.

Por su parte IBM(s.f.) presenta a API Connect como una solución a los bancos que necesitan producir grandes volúmenes de API's, para carteras digitales, asociaciones en tarjetas de crédito y programas de recompensas, analítica social e incluso para propósitos de geolocalización relacionados con cajeros automáticos. Asimismo, recomienda que la interfaz de gestor de API especializada puede ayudar a gestionar rápidamente la gestión completa y la monetización completa de innumerables API's.

1.2.2.8. Terminologías necesarias para API Connect.

En esta sección siguiente se describen los conceptos y la terminología utilizada en el empaquetado de IBM API Connect:

Planes y productos. IBM (2017) indica que los planes y productos son artefactos exclusivos de API Connect y explica que los proveedores utilizan Planes para controlar el acceso y gestión a las API, mientras que los Productos empaquetan el contenido tanto de las API como los Planes que las acompañan.

Para poner un API a disposición de un desarrollador de aplicaciones, tiene que estar incluida en al menos un Producto y en al menos un Plan.

Entre las funciones de un Plan tenemos las siguientes:

- Controlar las API que puede usar un desarrollador de aplicaciones.
- Poner a disposición una colección de operaciones procedentes de una o más API.
- Aplicar límites de velocidad a las API para diferenciar las ofertas.
- Implementar distintos límites de velocidad para especificar cuántas peticiones se permite realizar a una aplicación consumidora durante un determinado intervalo de tiempo.

En la Figura 3 se ilustra cómo puede utilizarse un Plan para exponer operaciones de una o más API y para definir límites de velocidad en ambos Planes y en operaciones individuales:

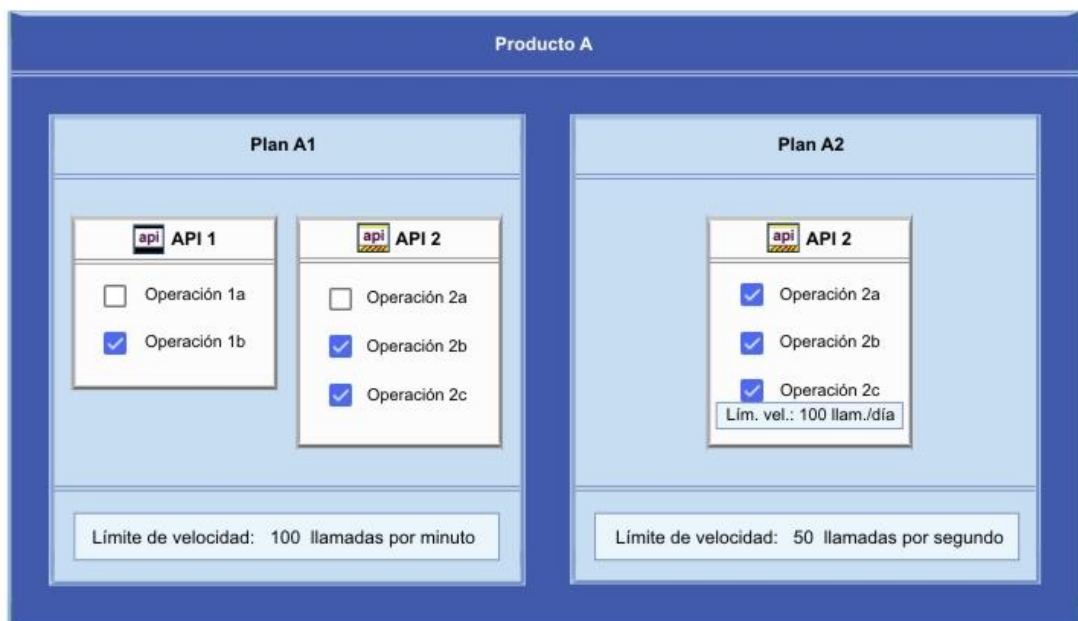


Figura 5. Relación entre Productos, planes y APIs (IBM, 2017).

La interpretación de la Figura 3 ilustra los conceptos siguientes:

- El Producto A contiene dos API y dos Planes.
- Tanto API 1 como API 2 están incluidas en el Plan A1.
- El Plan A2 solo incluye el API 2.
- La operación 2a del API 2 está excluida del Plan A1.
- Todas las operaciones del API 2 están incluidas en el Plan A2.
- Se han definido distintos límites de velocidad en ambos Planes a nivel de Plan.
- En el API 2 del Plan A2, también se ha definido un límite de velocidad específicamente para la operación 2c para sustituir el límite de velocidad definido a nivel de Plan.
- Los Productos se utilizan para gestionar el ciclo de vida de las API que contienen. Los estados del ciclo de vida del Producto pueden ser borrador, en desuso, retirado y archivado.

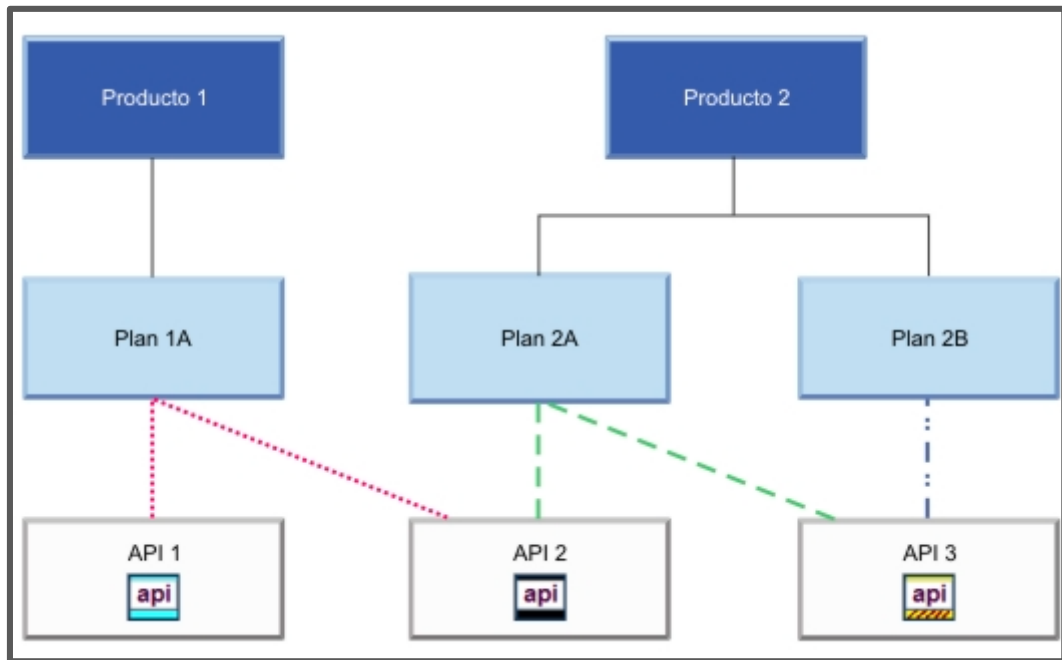


Figura 6. Jerarquía de Productos Planes y API (IBM, 2017).

Catálogos y Espacios. Un Catálogo contiene una colección de Productos. Los catálogos son destinos de transferencia a través de los cuales se publican los Productos en un Portal del desarrollador.

Los catálogos se usan para separar los Productos y las API permitiendo realizar pruebas antes de publicarlos en un Portal, existe un Catálogo de desarrollo para poder crear y probar las API, y otro Catálogo de producción para publicar las API que están listas para su uso externo. Cada catálogo tiene asociado un Portal de desarrollador para exponer los Productos publicados.

IBM API Connect incluye una funcionalidad que permite a los proveedores de API particionar un Catálogo en varios destinos de transferencia (staging o Espacios).

Organizaciones y usuarios. En API Connect existen dos tipos de organizaciones: proveedor y desarrollador IBM (2017) describe que entre las responsabilidades estándar de una organización de proveedores se incluyen:

- Posee el conjunto completo de permisos de acceso a las funciones de API Connect por lo que puede poner en servicio las API, y hacer un seguimiento de su aprobación empresarial.
- Desarrolladores de API que diseñan y desarrollan API y aplicaciones para las organizaciones de proveedores a las que pertenecen.
- Un administrador que gestiona los ciclos de vida de las API y publica las API para su descubrimiento y uso.
- Un gestor "de comunidad" que gestiona la relación entre la organización de proveedores y los desarrolladores de aplicaciones.
- Proporciona información relativa al uso del API y da soporte a los desarrolladores de aplicaciones.

Una organización de desarrolladores se denominan organizaciones de consumidores, solo posee aplicaciones de desarrollador y consume las API y aplicaciones producidas por la organización de proveedores.

Entre las responsabilidades estándar dentro de una organización de desarrolladores se incluyen:

- Añade desarrolladores de aplicaciones a la organización y visualiza los Productos y API, expuestos por la organización de proveedores y se suscribe a las API para usarlas en aplicaciones.
- Desarrolladores de aplicaciones que visualizan los Productos y API expuestos y se suscriben al uso de dichas API en aplicaciones.

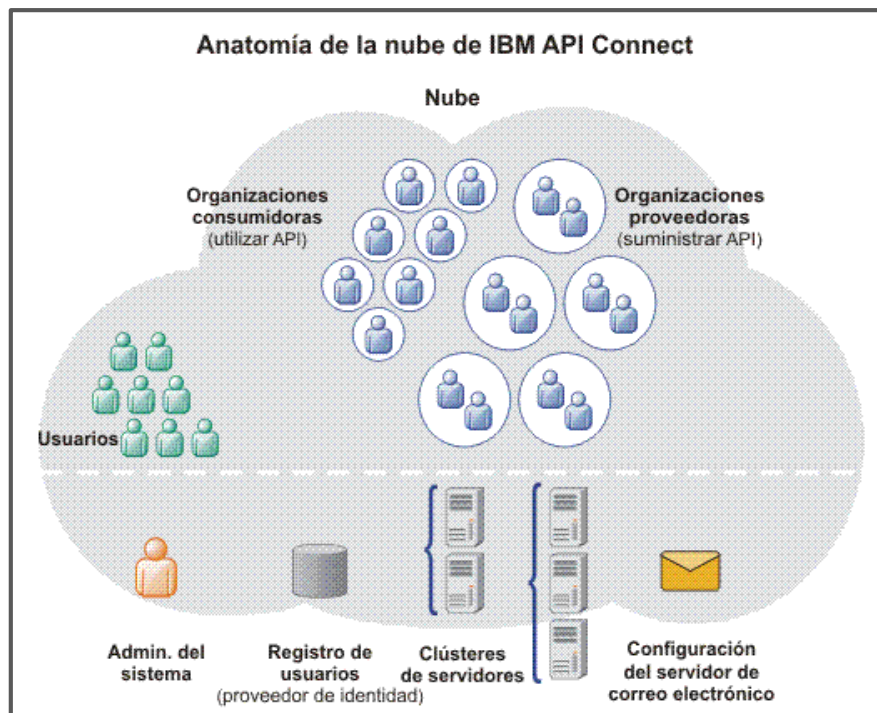


Figura 7. Organizaciones de proveedores, las organizaciones de desarrolladores y los usuarios (IBM, 2017)

Aplicaciones. Según IBM (2017) con las API asociadas también pueden crear aplicaciones compiladas con Node.js y tecnología Java. Al ser publicadas las aplicaciones de desarrollador invocan dichas API y aplicaciones. Dichas aplicaciones de desarrollador contienen código de cliente que accede a las API para interactuar con un servicio, sistema o contenido, estas suelen ser móviles o aplicaciones web que utilizan el protocolo HTTP.

1.2.3. Web Service

1.2.3.1. **Definición.**

Según Baquero (2015), un web service es una vía de intercomunicación e interoperabilidad entre máquinas conectadas en Red, por otro lado, Ionos (2020) sostiene que un web service facilita un servicio a través de Internet y se trata de una interfaz mediante la que dos máquinas (o aplicaciones) se comunican entre sí.

Para Lázaro (2018), un webservice es una colección de protocolos abiertos y estándares usados para intercambiar datos entre aplicaciones o sistemas. Las aplicaciones escritas en varios lenguajes de programación que funcionan en plataformas diferentes pueden utilizar web services para intercambiar información a través de una red.

Los tres autores refieren al webservice como el medio de comunicación entre los sistemas que permite su operatividad independientemente de su lenguaje de programación en el que fue desarrollado y plataforma en la que esté ejecutándose.

1.2.3.2. **Características.**

Según la Universidad de Alicante (2014), las características deseables de un Webservice son:

Accesible. Que pueda ser accesible a través de la Web con uso de protocolos de transporte estándares como HTTP y codificar los mensajes

en un lenguaje estándar que pueda conocer cualquier cliente que quiera utilizar el servicio.

Descriptivo. Debe contener una descripción de sí mismo con la finalidad que una aplicación podrá saber información acerca del web Service, tales como su función, interfaz, parámetros de entrada, entre otros, de manera que cualquier aplicación pueda utilizarlo sin que el usuario intervenga.

Localizado. Debe tener algún mecanismo que nos permita encontrar al web Service y se encuentre a disposición para otros usuarios.

1.2.3.3. Arquitectura.

Los servicios web presentan una arquitectura orientada a servicios permitiendo crear una definición abstracta del servicio.

Según JAVATPOINT(s.f.) existen tres roles en la arquitectura de servicios web:

Proveedor de servicio. Desde una perspectiva arquitectónica, es la plataforma la que aloja los servicios.

Solicitante de servicio. Solicitante de servicio es la aplicación que busca e invoca o inicia una interacción con un servicio. El navegador desempeña el papel de solicitante, impulsado por un consumidor o un programa sin interfaz de usuario.

Asimismo, en la arquitectura hay tres comportamientos diferentes:

Publicar. En la operación de publicación, se debe publicar una descripción del servicio para que un solicitante de servicio pueda encontrar el servicio.

Buscar. En la operación de búsqueda, el solicitante del servicio recupera la descripción del servicio directamente. Puede estar involucrado en dos fases diferentes del ciclo de vida para el solicitante del servicio:

En el diseño, es hora de recuperar la descripción de la interfaz del servicio para el desarrollo del programa.

Y, en el tiempo de ejecución, para recuperar el enlace del servicio y la descripción de la ubicación para la invocación.

Vincular. En la operación de vinculación, el solicitante del servicio invoca o inicia una interacción con el servicio en tiempo de ejecución utilizando los detalles de vinculación en la descripción del servicio para localizar, contactar e invocar el servicio.

JAVATPOINT(s.f.) describe que existen dos artefactos de los servicios web:

Servicio. Es una interfaz descrita por una descripción de servicio. La descripción del servicio es la implementación del servicio. Un servicio es un módulo de software implementado en plataformas accesibles a la red proporcionadas por el proveedor de servicios. Interactúa con un solicitante de servicio. En ocasiones también funciona como solicitante, utilizando otros Servicios Web en su implementación.

Descripción del servicio. Comprende los detalles de la interfaz y la implementación del servicio. Incluye sus tipos de datos, operaciones, información vinculante y ubicación de red. También puede categorizar otros metadatos para permitir el descubrimiento y uso por parte de los solicitantes de servicios. Puede publicarse en un solicitante de servicio o en un registro de servicios.

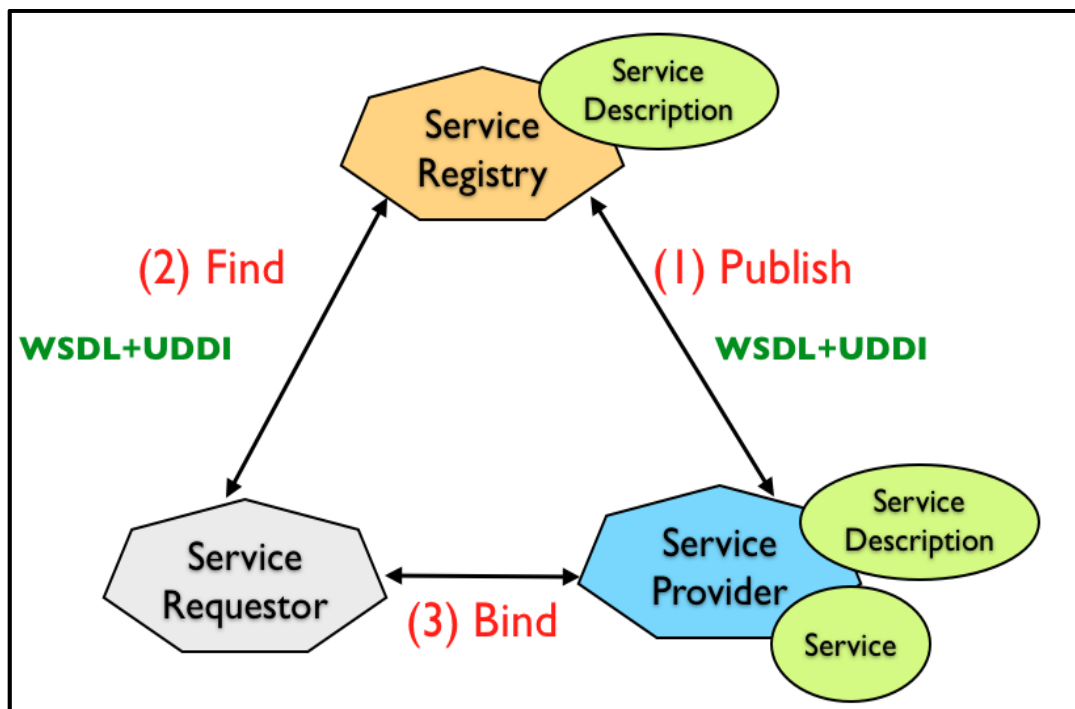


Figura 8. Diagrama de una arquitectura de Web Service orientada a servicios

1.2.3.4. Como funciona.

Según Lázaro (2018), los Web services funcionan de la siguiente manera:

1. El Service Provider genera el WSDL describe el Web Service y registra el WSDL en el directorio UDDI (Universal Descriptor, Discovery and Integration) o Service Registry.

2. El Service Requestor o la aplicación del cliente invoca al Web Service, para lograr esto utiliza el directorio UDDI para poder ubicar el Web Service.
3. El cliente, basándose en la descripción descrita por el WSDL, envía un request para un servicio particular al Web Service Listener, que se encarga de recibir y enviar los mensajes.
4. El Web Service analiza el mensaje del request e inicia la interacción con una operación particular en la aplicación y poder procesar el request. El resultado obtenido se escribe de nuevo, esta vez en forma de respuesta y es enviado al cliente.
5. El cliente analiza el mensaje de respuesta obtenido y lo interpreta.

1.2.3.5. Tipos.

1.2.3.5.1. Tipo SOAP.

Baquero (2015), describe SOAP como un protocolo que define cómo deben de realizarse las comunicaciones entre máquinas. SOAP usa XML como lenguaje de intercambio de datos con una estructura compleja que es capaz de albergar todo tipo de datos sobre la solicitud o respuesta generada.

Es un protocolo simple que facilita la comunicación entre dos objetos que pertenecen a procesos completamente diferentes, estos mensajes se forman por un grupo de etiquetas XML, cuenta con un lenguaje adicional que sirve para generar una interfaz del Web Service, basada en una

definición llamada “Descripción de lenguaje de Web Service” (WSDL por sus siglas en Ingles: “Web Services Description Language”), y que funciona como punto de entrada.

Para poder acceder al archivo WSDL o contrato se hace a través de un URL con terminación WSDL y se puede realizar una vez que este se encuentre publicado.

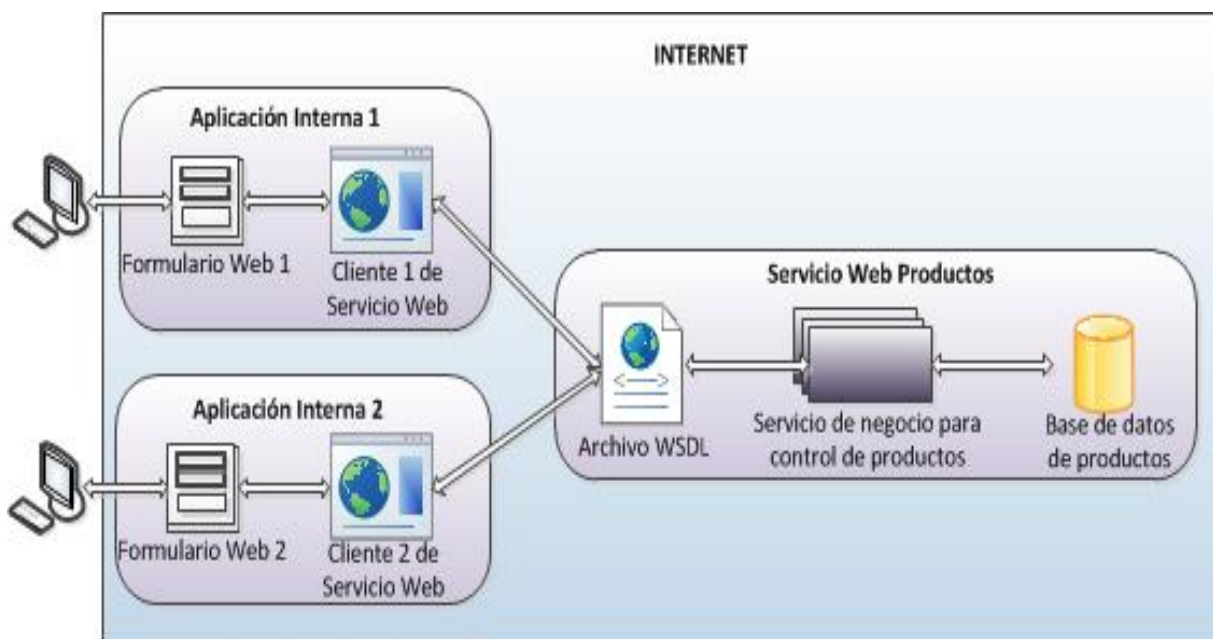


Figura 9. Concepto básico de servicio SOAP (Sandy, 2014)

1.2.3.5.2. Tipo REST.

Es un estilo de arquitectura para proporcionar estándares entre sistemas basados en el protocolo HTTP y permite obtener y generar datos y operaciones, devolviéndolos en formatos específicos tanto sea en XML o JSON. Actualmente, se opta más por el formato JSON, caracterizado por ser ligero, legible, sumamente simple comparándolo con el formato XML

que es más complicado de entender. La elección del formato radica en la cuestión de la lógica y necesidades según sea el proyecto.

BBVAOPEN4U (2016), menciona que REST es cualquier interfaz entre sistemas que use HTTP para obtener datos o generar operaciones sobre esos datos en todos los formatos posibles, como XML y JSON. Es una alternativa en auge a otros protocolos estándar de intercambio de datos como SOAP (Simple Object Access Protocol).

Asimismo, BBVAOPEN4U (2016), agrega que Las operaciones más importantes relacionadas con los datos en cualquier sistema REST y la especificación HTTP son cuatro: POST (crear), GET (leer y consultar), PUT (editar) y DELETE (eliminar).

Es por ello que surge una alternativa a SOAP.

1.2.3.5.3. SOAP VS REST.

Se debe tener en cuenta que SOAP y REST son servicios que siguen la misma arquitectura llamada SOA, la cual se basa en la integración de aplicaciones mediante Servicios, a pesar de tener los mismos principios, se pueden diferenciar de la siguiente manera:

Tabla 1

Comparación de Soap vs Rest.

Característica	SOAP	REST
DISEÑO	Protocolo oficial con requisitos específicos.	Conjunto de principios arquitectónicos que ofrece una implementación más flexible.
PROTOCOLO DE TRANSEFERENCIAS	Independiente del idioma, la plataforma y el transporte.	Requiere el uso de HTTP.
FORMATO DE MENSAJES	Usa XML para todos los mensajes.	Usa formatos de mensajes más pequeños (JSON, YAML, HTML, XML entre otros).
APRENDIZAJE	Su aprendizaje es complejo.	Curva de aprendizaje más pequeña
ENTORNO	Se recomienda para aplicaciones empresariales, entornos distribuidos.	Recomendado para APIS públicas, servicios WEB, servicios móviles.
VENTAJA	Su ventaja es alta seguridad, extensibilidad y estandarizada.	Su ventaja es Escalabilidad, mayor rendimiento, flexibilidad y facilidad de navegación.

Fuente: Autoría propia

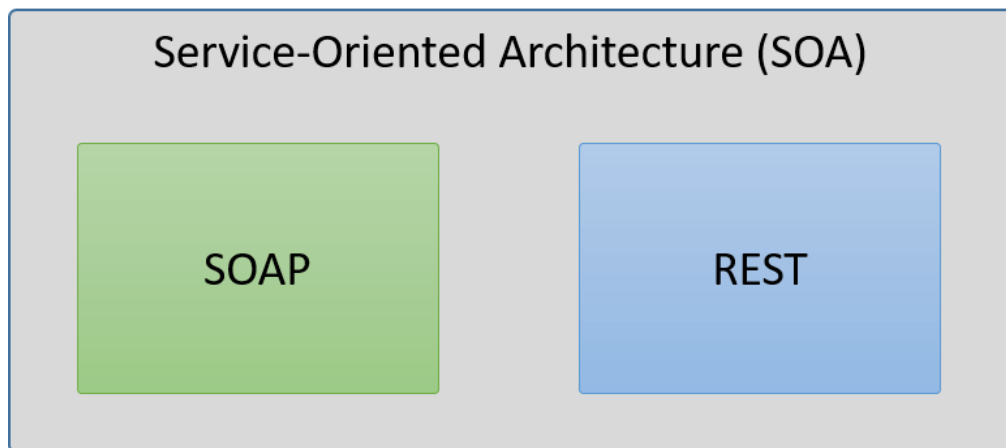


Figura 10. Servicios orientados a la arquitectura SOA (Blancarte, 2017)

1.2.3.6. Ventajas y Desventajas.

lonos(2020) La ventaja principal de los servicios web es que la comunicación no depende de una plataforma determinada, por lo que el cliente y el servidor apenas han de presentar rasgos en común para poder comunicarse. Para ello, la tecnología web service recurre a formatos estandarizados que interpretan todos los sistemas.

Pero, en estos formatos es donde encontramos una de las desventajas. Precisamente, XML es un formato más bien voluminoso que genera grandes paquetes de datos, lo que puede crear problemas en las conexiones de red lentas. Otra posibilidad que permite conectar a dos sistemas a través de Internet son las API web. Aunque, por lo general, son más rápidas, someten a cliente y servidor a especificaciones más concretas, con lo que la interoperabilidad se ve limitada.

1.2.4. Metodología de desarrollo de la plataforma

1.2.4.1. Definición de Extreme Programming (XP).

Es una metodología ágil la cual se basa en cuatro principios: simplicidad, comunicación, retroalimentación y valor. XP está orientada a reestructurar y a realizar pruebas, se diseña e implementan las pruebas antes de programar la funcionalidad, el programador debe crear sus test de unidad propios.

Este método es atribuido a Ron Jeffries, Kent Beck y Ward Cunningham. el objetivo principal de XP se basa en la construcción de software pequeños y medianos en donde existen requisitos que cambian con frecuencia o son de alto riesgo. XP busca que el cliente esté satisfecho durante todo el tiempo y mantenga su confianza en el producto. Asimismo, sugiere se considere el lugar de trabajo, un espacio amplio, si es posible sin divisiones (los programadores en el centro y los equipos individuales en la periferia). aprovechando como ventaja que en el espacio abierto se incrementa la comunicación y proporciona una agenda dinámica en el entorno del proyecto.

1.2.4.2. *Actividades de XP.*

Codifica. Es necesario representar y plasmar nuestras ideas e informaciones a través del código. En programación, el código nos ayuda a comunicar y expresa la interpretación del problema para hacer comunes las ideas; por tanto, para aprender y mejorar. En pocas palabras codificar porque sin código no hay programas.

Hacer pruebas. Las pruebas nos permiten verificar que nuestro trabajo funciona, permiten tener la oportunidad de saber que lo implementado es en realidad lo que se tenía en mente y son de suma importancia durante todo el proyecto en forma continua, se definen con los programadores inclusive antes de generar el código y que permitirá verificar que funcione correctamente.

Escuchar. Es necesario para un buen desarrollo el saber escuchar, si se realizaran pruebas tenemos que preguntar si lo obtenido es lo deseado, y también que preguntar quién necesita la información. Escuchar a nuestros clientes nos ayudará a entender los problemas de su negocio, debemos de tener una escucha activa explicando lo que es fácil y difícil de obtener, y la realimentación entre ambas partes nos ayudarán a entender los problemas con mayor claridad. Debemos escuchar, porque si no escuchamos no sabemos que codificaremos o probaremos.

Diseñar. El diseño nos ayuda a organizar una estructura para la lógica del sistema, un buen diseño permite que el sistema crezca con cambios en un solo lugar. Se recomienda que los diseños no sean complejos; pero, si algún componente del sistema es complejo debería separarse en varias partes, el diseño es muy importante así que en caso se encuentren defectos o diseños mal elaborados, deben ser corregidos en la brevedad. Debemos diseñar para poder codificar, probar y escuchar indefinidamente.

1.2.4.3. Prácticas básicas de XP.

La Metodología XP tiene el siguiente conjunto de prácticas:

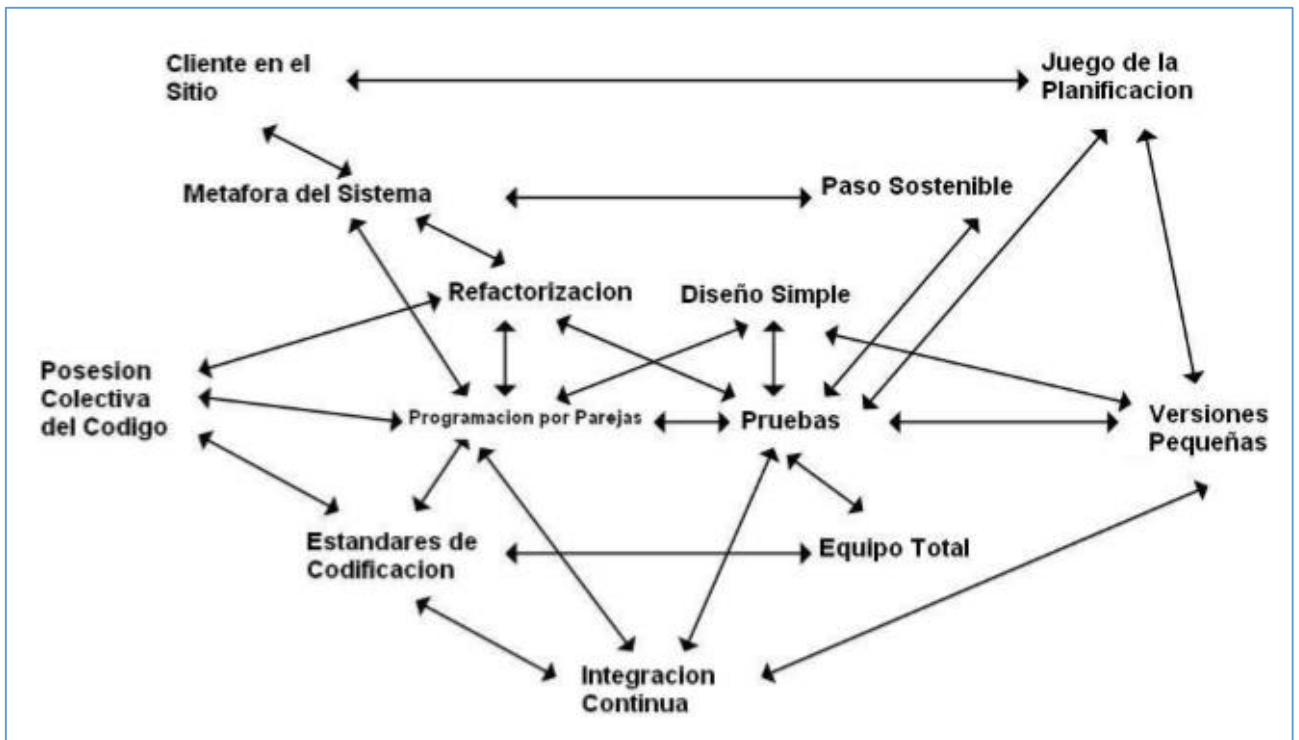


Figura 11. Conjunto de Prácticas de la metodología XP (Universidad de Buenos Aires, 2016)

1.2.4.4. Ciclos de XP.

La metodología XP se basa en procesos interactivos e incrementales del desarrollo, en un período de tiempo se realizan un grupo de funcionalidades determinadas que se representan en un conjunto de historias de usuarios.

Las fases del ciclo de vida XP, se subdividen:

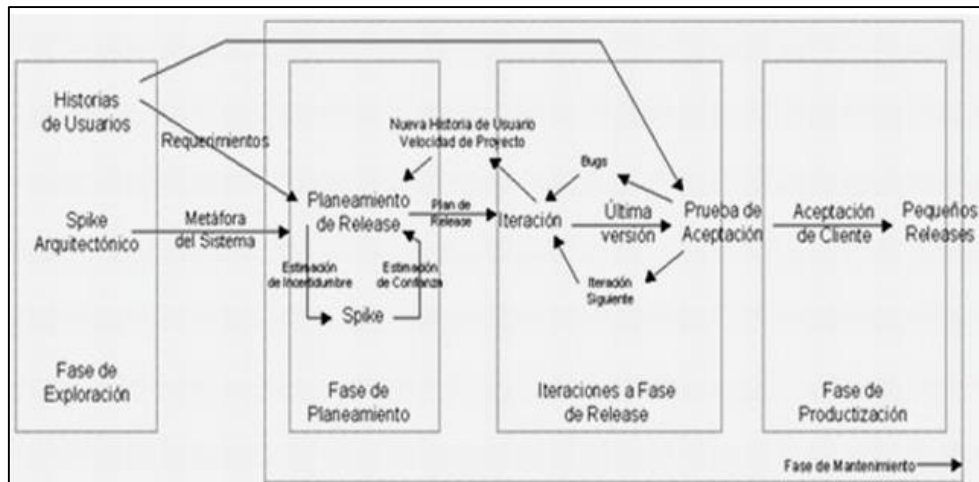


Figura 12. Fases del ciclo de vida de la metodología XP (Hurtado y Bastiarrica, 2016)

1.2.4.5. Actores y Responsabilidades de XP.

Programador(Programmer).

- Es la pieza fundamental en desarrollos XP.
- Tiene mayor responsabilidad que en otros modos de desarrollo.
- Es el responsable del código y las decisiones técnicas que se toman.
- Es el responsable del diseño (basado en la simplicidad y refactorización).
- Es el responsable de la integridad del sistema (ejecución de pruebas).
- Tiene alta capacidad de comunicación.
- Acepta las críticas (utiliza código colectivo).
- En la metodología XP, los programadores son los que diseñan, programan y realizan las pruebas del sistema.

Cliente(Customer).

- Pieza básica en desarrollos de la metodología XP.
- Define las especificaciones, para saber qué construir y cuándo hacerlo.
- Influye sin controlar.
- Tiene confianza en el equipo de desarrollo.
- Define las pruebas funcionales.

Encargado de pruebas(Tester).

- Apoya al cliente en la preparación y la realización de las pruebas funcionales en el sistema.
- Es el responsable de ejecutar las pruebas funcionales y evidencia los resultados

Encargado de seguimiento (Tracker)

- Es llamado Metric Man.
- Se encarga de observar sin molestar.
- Conserva los datos históricos.
- Es responsable de recoger, analizar y publicar la información sobre el estado del proyecto sin afectar demasiado el proceso.

- Es responsable de supervisar el cumplimiento de las estimaciones en cada una de las iteraciones.
- Tiene control del estado de las pruebas funcionales, las responsabilidades aceptadas, los errores que han sido reportados y las pruebas añadidas por los errores encontrados.

Entrenador (Coach)

- Es el experto en la metodología XP.
- Es el responsable del proceso en todo su conjunto.
- Identifica las desviaciones del proyecto y reclama la atención sobre ellas.
- Guía al equipo de forma indirecta.
- De ser necesario interviene directamente.
- Es el encargado de atajar rápidamente el problema.

Consultor

- Apoya al equipo en temas puntuales y cubre sus necesidades.
- Es el jefe del Proyecto.
- Favorece la relación entre los usuarios y los desarrolladores.
- Asegura que el equipo alcance sus objetivos.



Figura 13. Los roles y responsabilidades de XP (Solcataleya, 2016)

1.2.4.6. Artefactos de XP.

Los artefactos de XP son:

Historias de Usuario. Representan una breve descripción del comportamiento del software, especificando sus características funcionales y no funcionales emplea terminología del cliente, siendo fáciles de interpretar, las historias permiten a los desarrolladores hacer estimaciones de tiempo y planes del lanzamiento, pueden ser creadas durante conversaciones con los demás actores sobre nuevas funcionalidades o mejoras del proyecto. Sustituyen a grandes documentos de requisitos.

Historia de Usuario	
Número:	Nombre Historia de Usuario:
Modificación (o extensión) de Historia de Usuario (Nro. y Nombre):	
Usuario:	Iteración Asignada:
Prioridad en Negocio: (Alta / Media / Baja)	Puntos Estimados:
Riesgo en Desarrollo: (Alto / Medio / Bajo)	Puntos Reales:
Descripción:	
Observaciones:	

Figura 14. Modelo propuesto para una historia de usuario.

Tareas de Ingeniería.

Tarea de Ingeniería	
Número Tarea:	Historia de Usuario (Nro. y Nombre):
Nombre Tarea:	
Tipo de Tarea : Desarrollo / Corrección / Mejora / Otra (especificar)	Puntos Estimados:
Fecha Inicio:	Fecha Fin:
Programador Responsable:	
Descripción:	

Figura 15. Modelo propuesto para una tarea de ingeniería.

Tarjetas CRC (Clase - Responsabilidad – Colaborador). Las tarjetas CRC se fraccionan en tres secciones que contienen la información del nombre de la clase, sus responsabilidades y sus colaboradores. En la siguiente imagen observamos cómo se distribuye esta información.

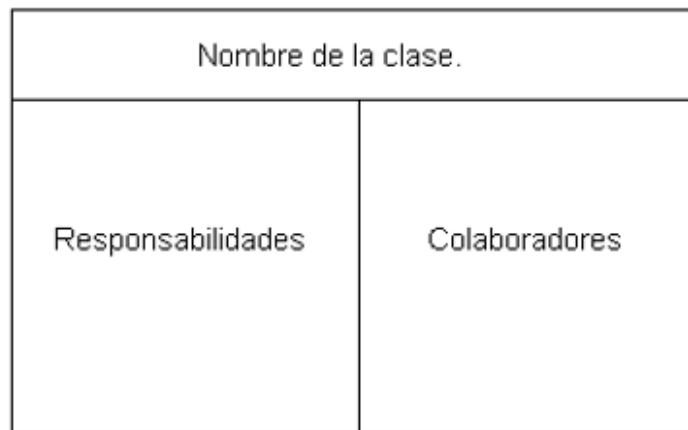


Figura 16. Modelo de tarjeta CRC (Universidad de Buenos Aires, 2016).

1.2.5. Área de integración de Interbank

1.2.5.1. Banco Interbank.

El Banco Internacional del Perú S.A.A. - Interbank, se fundó un primero de mayo de 1897, y hasta la fecha Interbank es una de las principales instituciones financieras del país.

Interbank está enfocado en brindar un servicio de calidad y productos llenos de innovación a más de 2 millones de clientes.

Tiene como Visión “Ser el mejor banco a partir de las mejores personas” y Como propósito fundamental ser una guía a cada peruano hasta alcanzar sus sueños.

Cuenta con plataformas de productos Digitales como:

Tunki. Es una Aplicación móvil que permite a las personas enviar dinero directo a sus contactos y pagar sus compras de manera práctica y rápida.

Aplicación Móvil Interbank. Es una herramienta digital solida entre Interbank y sus clientes.

Smart. Es un reporte de gastos mensual que ayuda a sus clientes a hacer una clasificación de sus gastos y compararlos periódicamente para mantener el orden de sus finanzas.

Entre otras herramientas Digitales, las cuáles hacen uso de servicios Web Services dentro de una plataforma de servicios de Integración.

1.2.5.2. Estructura organizacional actual de la empresa.

El Banco está conformado por aproximadamente 7,000 colaboradores; cuenta con alrededor de 270 agencias en Lima y Provincias y atiende más de 5'000,000 de clientes entre personas naturales y jurídicas (empresas). Está conformado por siete Vicepresidencias Ejecutivas quienes lideran en total 55 divisiones (incluyendo dos Gerencias Centrales).

En el siguiente diagrama se muestra la estructura organizacional del Banco:

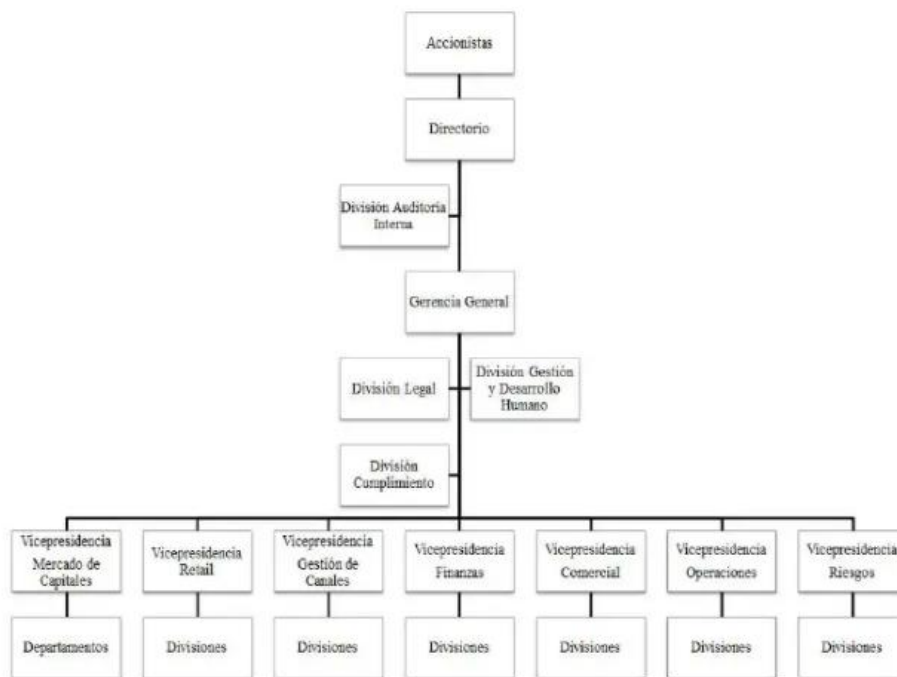


Figura 17. Organigrama de Interbank (Murguía, 2016).

1.2.5.3. **Plataforma de Integración en el banco Interbank.**

Los servicios de Integración tienen un modelo de arquitectura de software que gestiona la comunicación entre múltiples servicios web, se enfoca en resolver el problema que surge cuando los servicios web dentro de una organización se multiplican lo que hace necesario desarrollar conectores que permitan comunicar las diferentes aplicaciones.

Interbank hace uso de diferentes servicios Webservice por lo que hace uso de esta Herramienta. Pero, estos servicios han tenido una gran evolución que incluyen entre otros una estrategia IT en el desarrollo de APIs; pero, son pocas las organizaciones que están adaptándose a esta innovación.

1.3. Definición de términos básicos

1.3.1. ALMACENAMIENTO CACHÉ

Es una capa de almacenamiento auxiliar o temporal de datos solicitados con mayor frecuencia. su objetivo es aumentar el desempeño para recuperar datos y evitar entrar a la capa subyacente de almacenamiento y que el proceso se ponga más lento.

1.3.2. BASE DE DATOS

Una base de datos es una colección de datos que se estructuran y relacionan entre si y se encuentran almacenados para su uso posterior, En la informática se refiere a un sistema conformado por una agrupación de datos que se almacenan en discos para poder tener acceso a ellos, así como un conjunto de programas que manipulen ese conjunto de datos.

1.3.3. DATA POWER

Es un producto de IBM creado para cubrir las necesidades de seguridad, control e integración en las diversas plataformas como web, móviles, API, SOA, B2B (business to business) y la nube, DataPower tiene una pasarela que protege, controla y acelera las cargas de trabajo en DMZ y zonas de confianza. Asimismo, tiene una pasarela segura a la solución IBM API Connect con la que se puede controlar, proteger, supervisar, publicar y gestionar APIs.

1.3.4. HTTP

HTTP (HyperText Transfer Protocol) es un conjunto de reglas que un servidor sigue cuando se trata de la transmisión de archivos para la comunicación como imágenes, videos, audio y otros archivos a través de la World Wide Web (WWW). Cuando un usuario hace uso de un navegador, ya está haciendo uso de un HTTP. Básicamente, es un protocolo de transferencia de hipertexto que se ejecuta a través de la parte superior de la suite de protocolos TCP/IP. Fue diseñado para la comunicación entre los navegadores y servidores web; aunque puede ser utilizado para otros propósitos también.

1.3.5. HTTPS

HTTPS (Hypertext Transfer Protocol Secure), este protocolo de comunicación es la versión segura del protocolo HTTP, por el cual se envían los datos entre el navegador y el sitio web.

El sistema HTTPS utiliza un cifrado basado en la seguridad de textos SSL/TLS para crear un canal cifrado, con el fin de evitar ataques que puedan interceptar la transferencia de datos en las conexiones, en el caso que el atacante llegará a romper la conexión no podrá resolver los datos cifrados.

1.3.6. IBM INTEGRATION BUS

IBM Integration Bus direcciona, transforma y enriquece la información empaquetada como los mensajes de una ubicación cualquiera a cualquier otra ubicación entre diferentes aplicaciones, desde grandes sistemas típicos hasta dispositivos sin nombre, como sensores en interconexiones.

El producto da soporte a una amplia gama de protocolos como: HTTP y HTTPS, WebSphere MQ, servicios web, File, JMS 1.1 y 2.0, Enterprise Information Systems y TCP/IP.

Da soporte a una amplia gama de formatos de datos como formatos binarios, XML y estándares del sector. También puede definir sus propios formatos de datos.

Da soporte a muchas operaciones como el direccionamiento, enriquecimiento, la transformación, seguimiento, colección, la distribución, correlación y detección.

1.3.7. URL O URI

URI o URL: Uniform Resource Identifier, es una dirección única asignada a un recurso de internet, nos permite encontrar aquello que buscamos en internet, puede hacer referencia sitios web, texto, imágenes, videos, etc.

1.3.8. WINSCP

WinSCP es una aplicación de Software Libre. WinSCP es un cliente SecureFTP gráfico para Windows que emplea sobre SSH-1 y SSH-2. Su función principal es permitir la transmisión segura ya que utiliza métodos criptográficos integrados de archivos entre dos sistemas informáticos, el local y uno remoto que ofrezca servicios SSH.

1.3.9. WSDL

WSDL es un documento XML para describir un servicio web. Una definición WSDL indica la composición de una solicitud de web service y da una descripción de la interfaz proporcionada por el proveedor del web

service. Las definiciones WSDL están divididas en varias secciones, en las que se especifican la interfaz lógica y detalles físicos del web service como el enlace del protocolo de red y formato de mensaje utilizado.

1.3.10. XML

XML (Extensible Markup Language) son las siglas del Lenguaje de Etiquetado Extensible este es un lenguaje de etiquetas es simplificado, adaptado a Internet, este lenguaje contiene paquetes de información que están delimitados por dos etiquetas.

Estos conjuntos de códigos se pueden utilizar en el análisis de la lectura o datos de textos creados por personas o computadoras.

CAPÍTULO II: METODOLOGÍA DE DESARROLLO DEL TRABAJO PROFESIONAL

2.1. Delimitación temporal y espacial del trabajo

2.1.1. Delimitación temporal

El presente estudio tiene un horizonte temporal comprendido en el año 2019, dividido en dos etapas:

Primera Etapa. Desarrollado en el horizonte de tiempo comprendido entre los meses de noviembre del 2019 y mayo20 del 2020, tiempo en el que se pudo realizar el diagnóstico del problema, la fase de exploración y planificación.

Segunda Etapa. Se desarrollará tomando un horizonte de tiempo comprendido desde mayo hasta setiembre del 2020, donde se incluyen la elaboración del marco teórico, la metodología de desarrollo, conclusiones y recomendaciones.

2.1.2. Limitación Espacial

Banco Internacional del Perú S.A.A. (Interbank Perú), es una de las principales instituciones financieras del Perú. Está enfocado en brindar productos innovadores y un servicio conveniente y ágil a sus más de 2 millones de clientes, el cual se encuentra ubicado en la Av. Carlos Villarán 140, en el distrito de La Victoria-Lima, Perú



Figura 18. Sede Central del Banco Internacional del Perú INTERBANK (Oficina de consultoría en el sector de la ingeniería civil en Lima Perú, 2000).

2.2. Determinación y análisis del problema

Los Webservice se definen como un sistema de software diseñado para dar soporte a la interacción de máquina a máquina a través de una red, estos realizan una tarea o un conjunto de tareas específicas, y se describe mediante una notación estándar XML llamada WSDL (Web Services Description Language), la cual proporciona todos los detalles necesarios para interactuar con el servicio, incluidos los formatos de mensaje que detallan tanto las operaciones como los protocolos de transporte y la ubicación.

El Banco Internacional del Perú S.A.A. (Interbank Perú) es una empresa que opera principalmente en el sector Seguros y Banca, posicionada como una de las principales instituciones financieras del Perú y está

enfocado en brindar productos innovadores y un servicio ágil y de calidad a más de 2 millones de clientes.

Interbank para la realización de sus diversas operaciones bancarias como altas de un cliente, altas de tarjetas, apertura de cuentas, transferencias de dinero, consultas de saldos, etc. En su plataforma de integración utiliza los Webservice como solución de intercambio de datos entre sus aplicaciones distribuidas que se ejecutan en diferentes sistemas operativos y arquitecturas.

Estos Web services basan su comunicación bajo el protocolo SOAP (Simple Object Access Protocol), mediante metalenguaje XML que le permite ser utilizado en cualquier lenguaje; sin embargo, es un formato pesado, tanto en tamaño como en procesamiento; además de la complejidad de lectura. Asimismo, si se desea modificar algo en el servidor esto impacta de una forma negativa en el tiempo, debido a que se deben realizar varias modificaciones al código y realizar despliegues entre los ambientes pre-productivos y productivos.

Las compañías de todos los sectores, especialmente los del sector financiero están siendo más competitivas, por estar en medio de un proceso de transformación digital, lo cual ha llevado a la necesidad de creación de servicios web más ligeros y simples, fácilmente visibles y utilizables, con una mejora en la solicitud de procesamiento y una

devolución de los datos necesarios, esta solución la vienen brindado las APIS (interfaces de programación de aplicaciones).

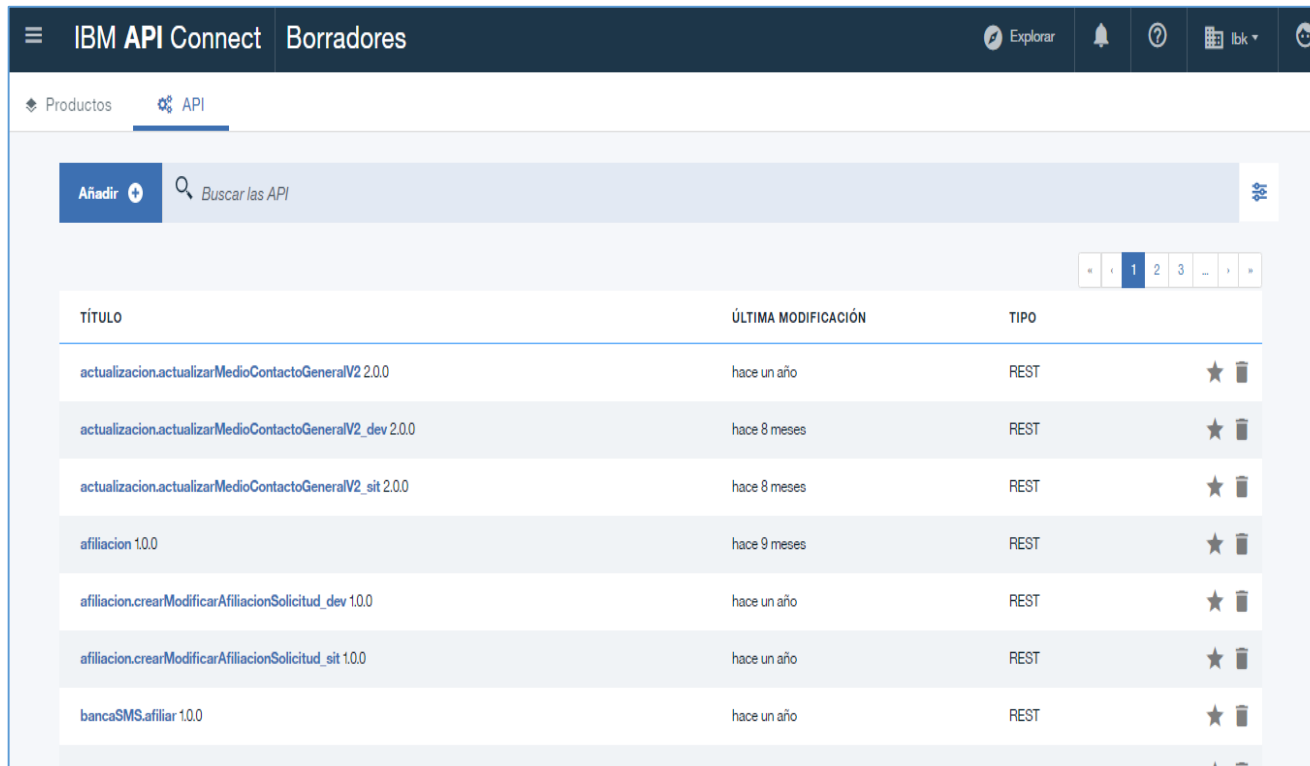
Para que el banco Interbank pueda mantenerse a la vanguardia de la tecnología, es necesaria la implementación de una herramienta API para la migración y mejora de los desarrollos de la plataforma de integración de servicios web service, que le permita mejorar el performance en su comunicación de datos, organizar su comunicación interna para dar apoyo a nuevos proyectos innovadores de manera uniforme, reducir costes de mantenimiento y despliegue de servicios, incrementar la agilidad de los procesos de transformación.

2.3. Modelo de solución propuesto

2.3.1. Recursos para emplear

2.3.1.1. *Herramienta API MANAGER.*

Es una plataforma de desarrollo que nos permite crear y gestionar servicios API del tipo REST.



The screenshot displays the IBM API Connect interface. At the top, there is a navigation bar with 'IBM API Connect' and 'Borradores' (Drafts). Below this, there are tabs for 'Productos' (Products) and 'API'. A search bar is present with the text 'Buscar las API' and an 'Añadir' (Add) button. The main content area shows a table of API products with columns for 'TÍTULO', 'ÚLTIMA MODIFICACIÓN', and 'TIPO'. Each row includes a star icon for favorites and a trash icon for deletion. The table lists several API endpoints, including 'actualizacion.actualizarMedioContactoGeneralV2 2.0.0' and 'afiliacion.1.0.0'.

TÍTULO	ÚLTIMA MODIFICACIÓN	TIPO
actualizacion.actualizarMedioContactoGeneralV2 2.0.0	hace un año	REST
actualizacion.actualizarMedioContactoGeneralV2_dev 2.0.0	hace 8 meses	REST
actualizacion.actualizarMedioContactoGeneralV2_sit 2.0.0	hace 8 meses	REST
afiliacion.1.0.0	hace 9 meses	REST
afiliacion.crearModificarAfiliacionSolicitud_dev 1.0.0	hace un año	REST
afiliacion.crearModificarAfiliacionSolicitud_sit 1.0.0	hace un año	REST
bancaSMS.afiliar 1.0.0	hace un año	REST

Figura 19. Interfaz de la herramienta API Manager (Elaborado por la investigadora).

2.3.1.2. Herramienta IBM DataPower Gateway.

Esta herramienta proporciona seguridad, control, integración y acceso optimizado a un rango completo de cargas.

Asimismo, se hará uso de su complemento de Dashboard que nos proporciona un panel de operaciones de DataPower donde se pueden ver los detalles de transacciones en una o más pasarelas DataPower y ayuda en la identificación de problemas de cumplimiento de seguridad de errores de servicio.

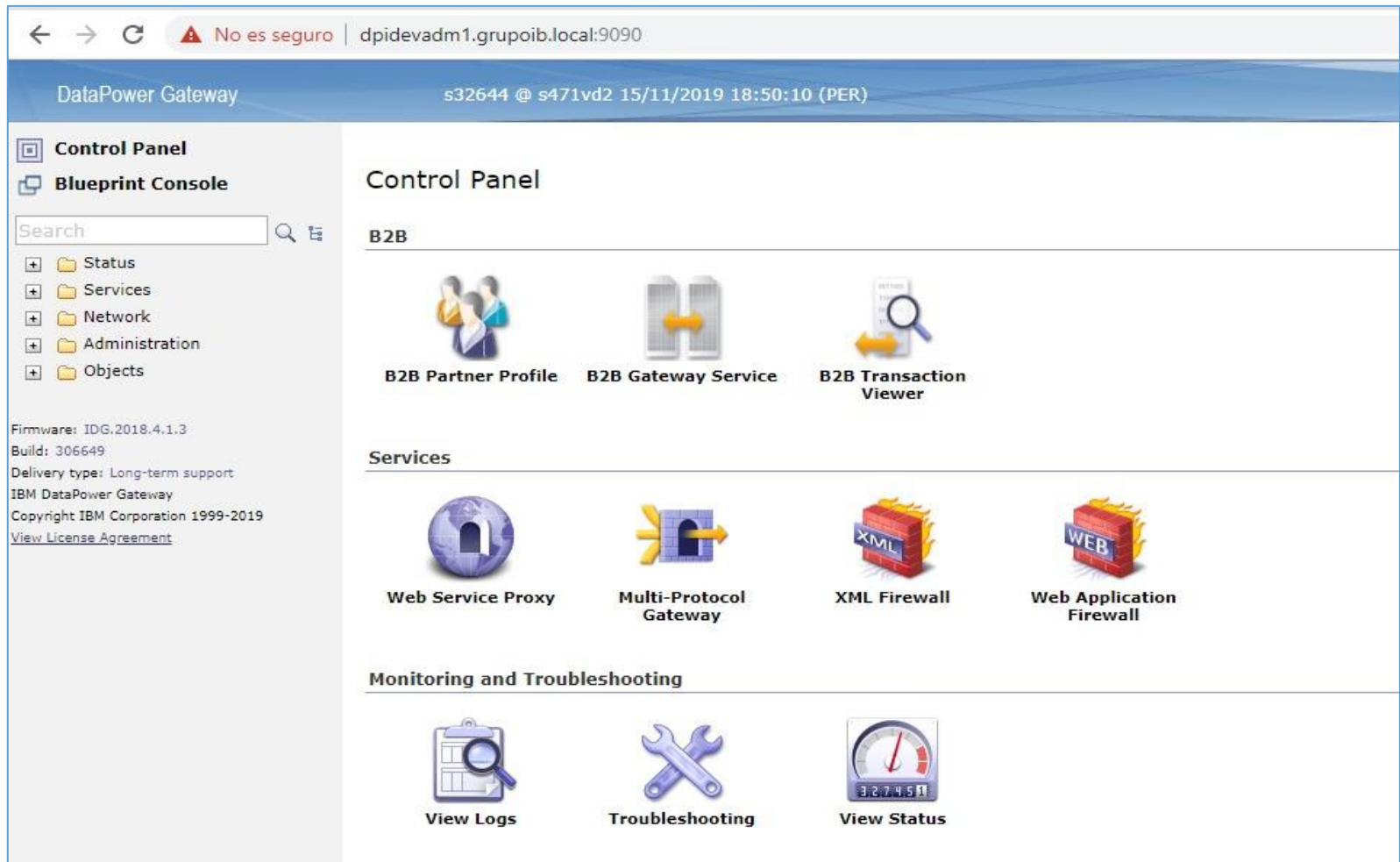


Figura 20. Interfaz de la herramienta Data Power Gateway (Elaborado por la investigadora).

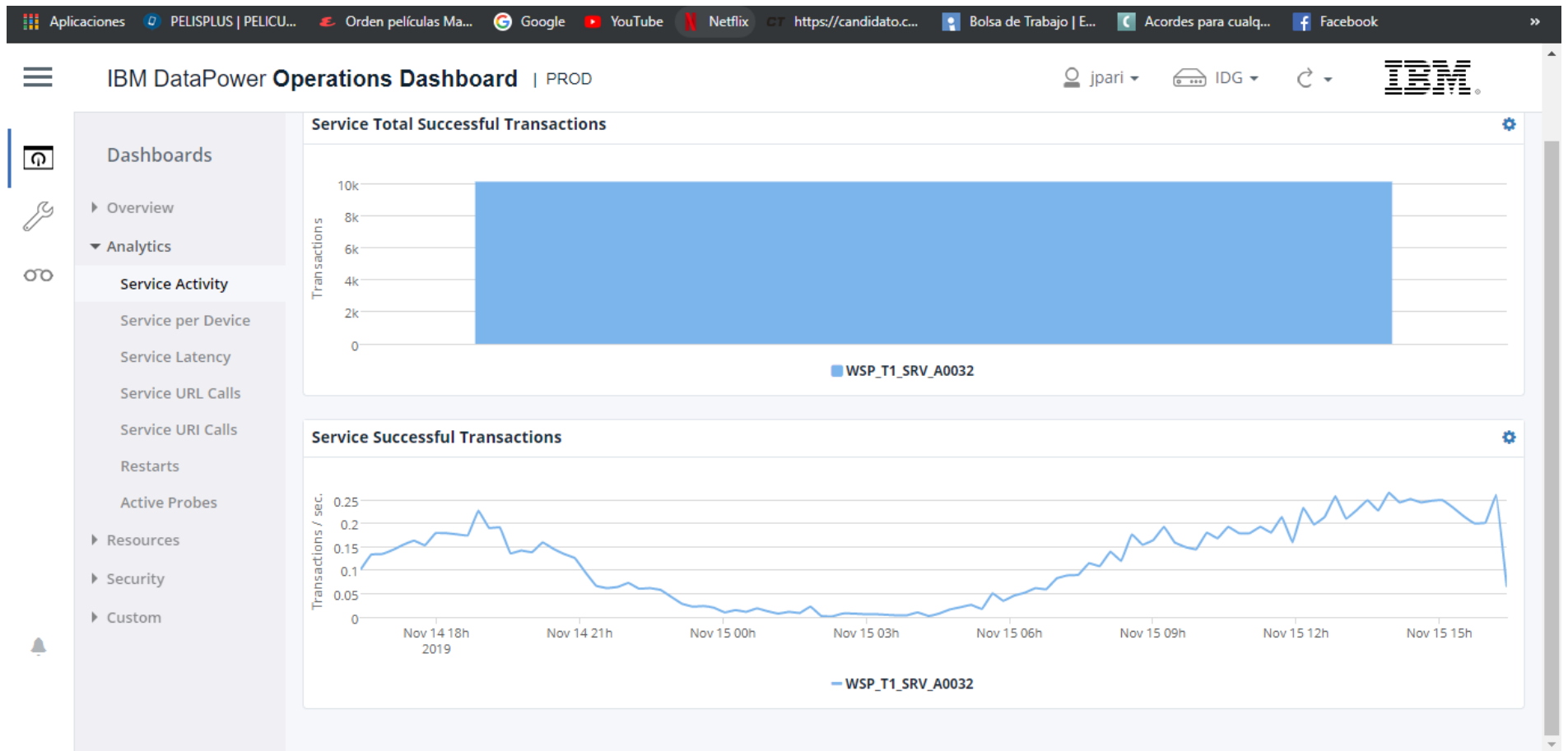


Figura 21. Interfaz de la herramienta Data Power operations Dashboard (Elaborado por la investigadora).

2.3.1.3. Herramienta JMETER.

Es una herramienta de Apache que nos permite realizar las pruebas de rendimiento para tener una medición de los tiempos de respuestas de los web services.

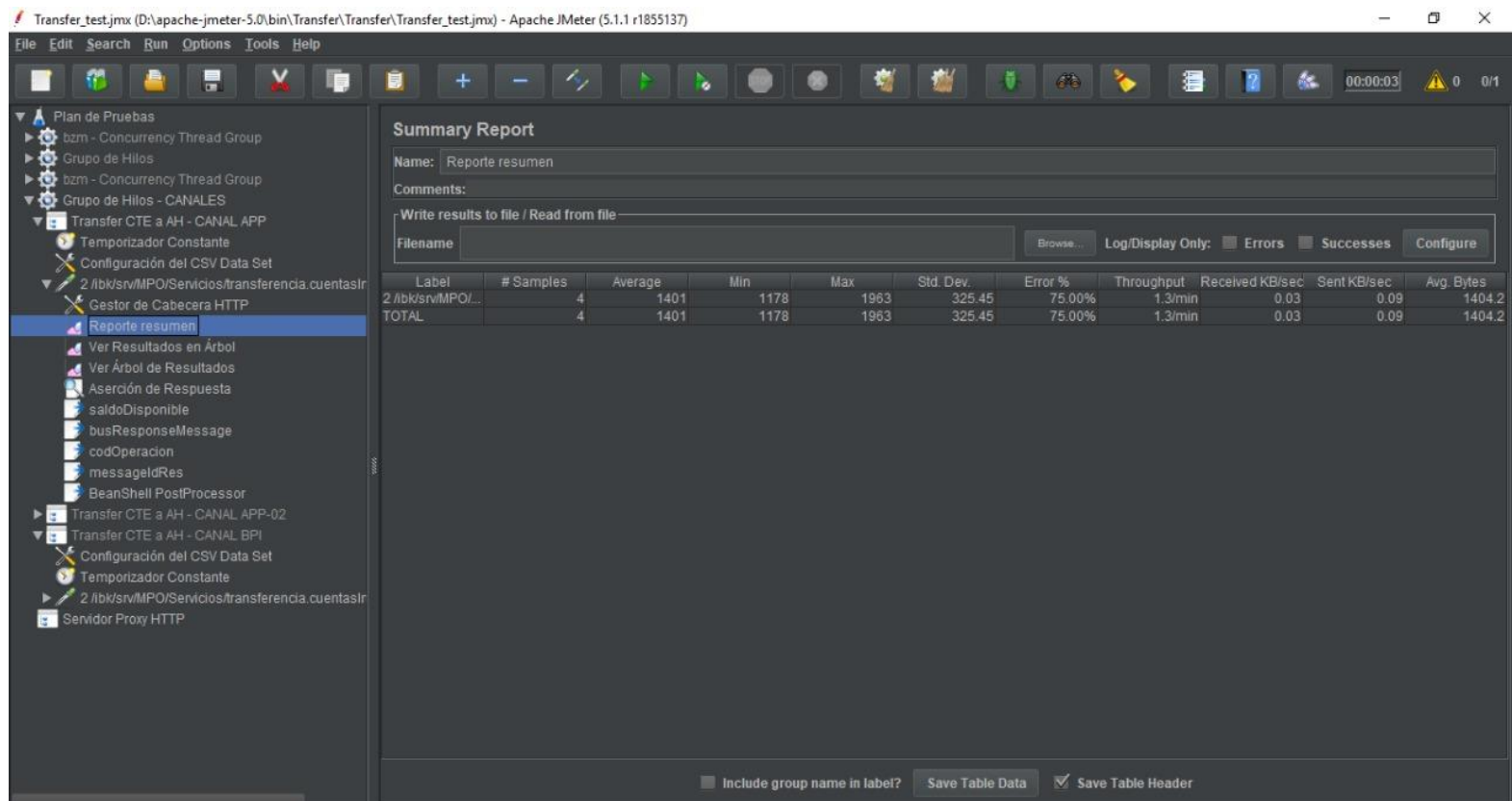


Figura 22. Interfaz de la herramienta Jmeter (Elaborado por la investigadora).

2.3.1.4. Herramienta Postman.

Es una herramienta Web que permite realizar peticiones a cualquier API, se envían los parámetros necesarios y se analiza la respuesta del Web Service.

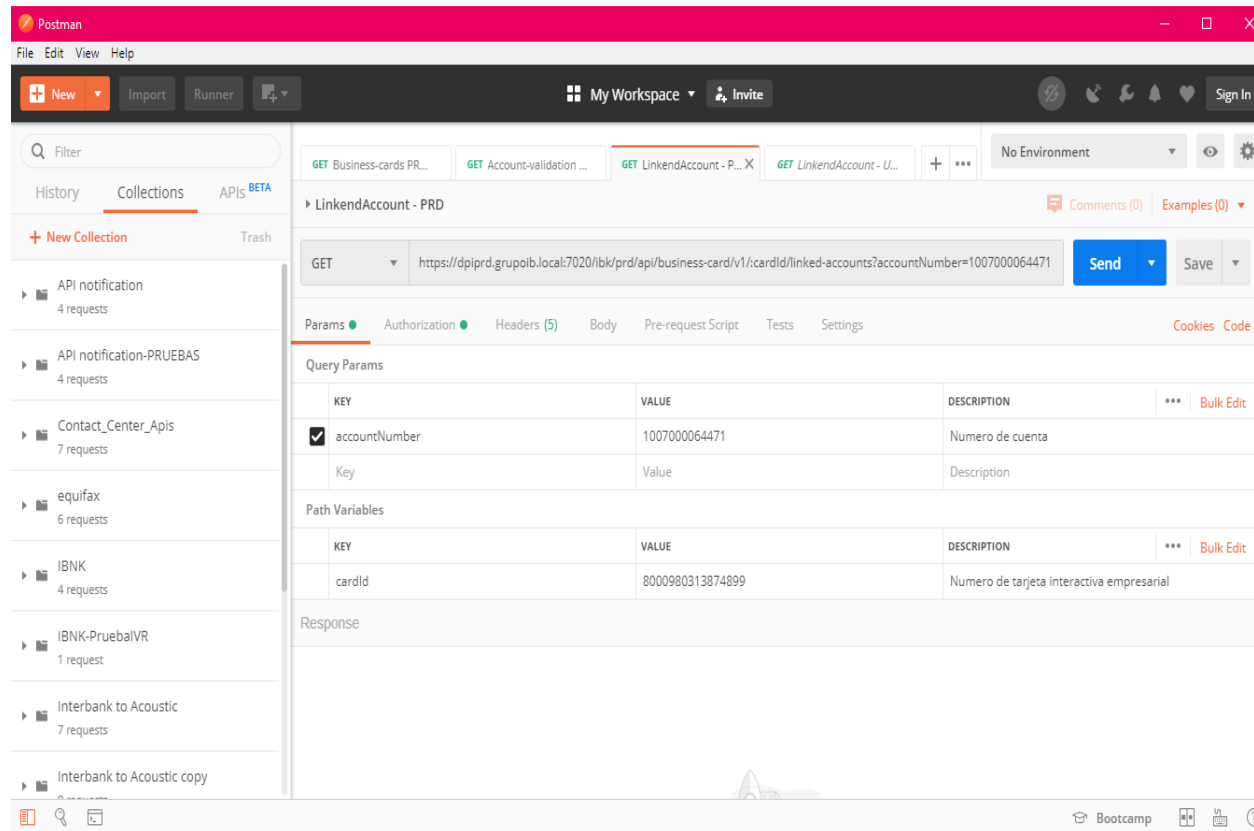


Figura 23. Interfaz de la herramienta Postman (Elaborado por la investigadora).

2.3.1.5. Herramienta SOAPUI.

Esta herramienta nos permite realizar las pruebas a los Web Services orientados a servicios SOAP, se envían los parámetros necesarios y se analiza la respuesta del Web Service

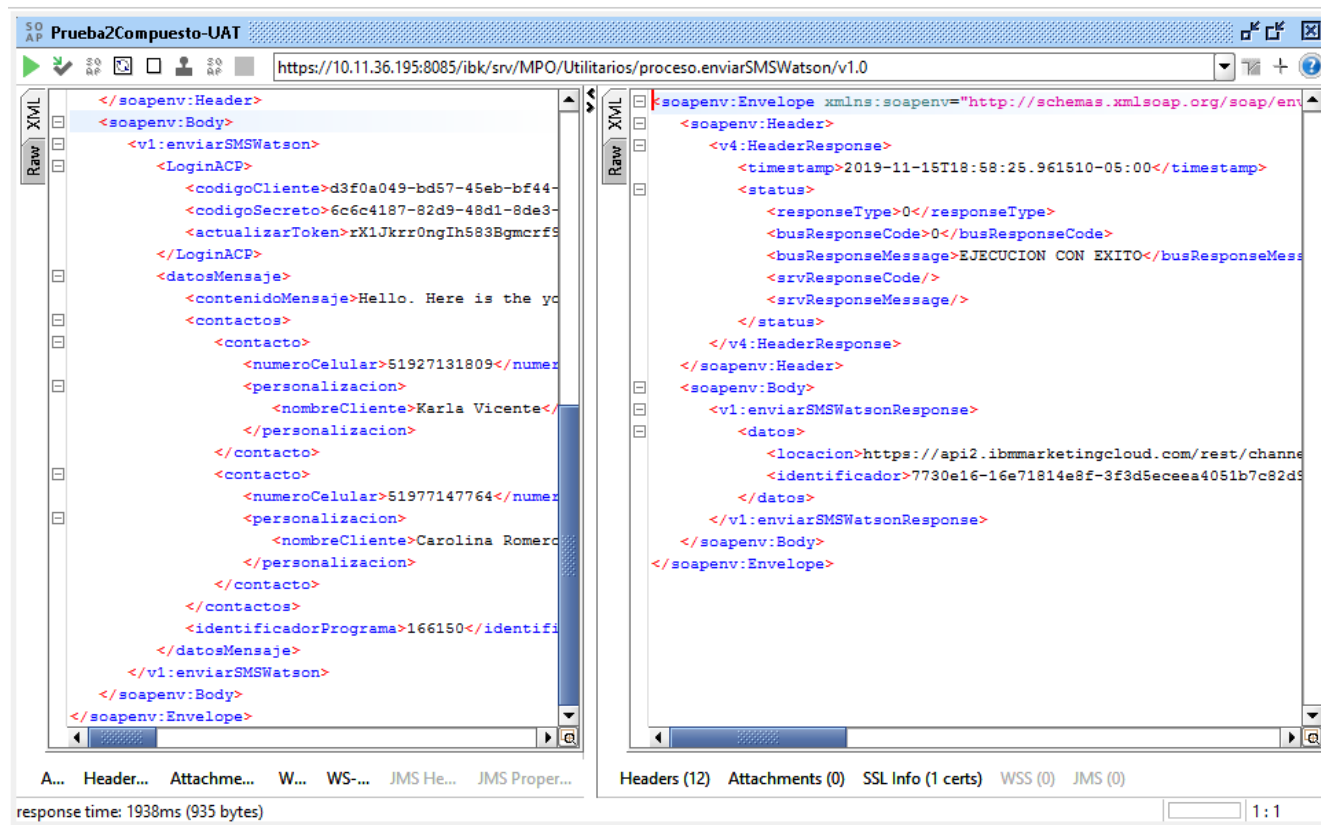


Figura 24. Interfaz de la herramienta SOAPUI (Elaborado por la investigadora).

2.3.1.6. Herramienta Winscp.

Su función principal es facilitar la transferencia de archivos entre nuestro servidor local y el servidor principal remoto.

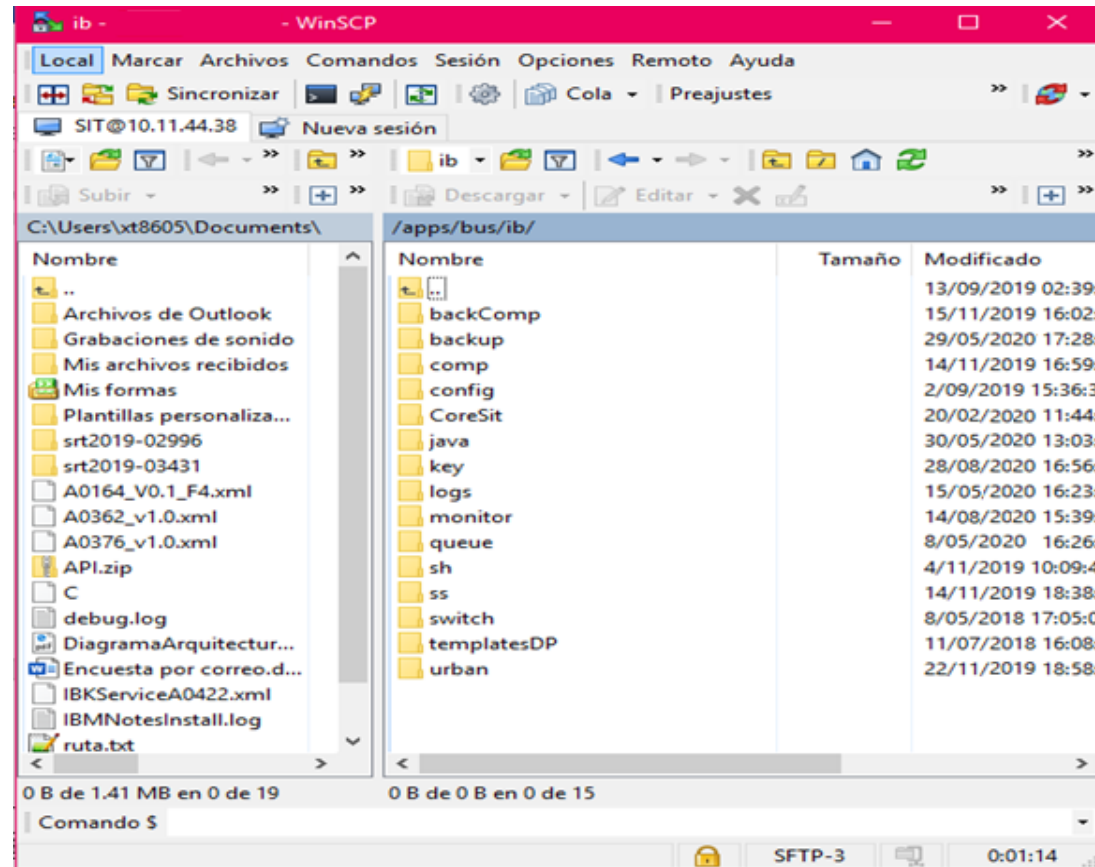


Figura 25. Interfaz de la herramienta WINSCP (Elaborado por la investigadora).

2.3.2. Procedimientos

2.3.2.1. Cronograma de Análisis.

Tabla 2 Cronograma de actividades

Actividad	Semana 1	Semana 2	Semana 3	Semana 4	Semana 5	Semana 6	Semana 7	Semana 8	Semana 9	Semana 10	Semana 11	Semana 12	Semana 13	Semana 14	Semana 15	Semana 16
Diagnóstico del problema																
Fase de Exploración																
Fase de Planificación																
Fase de Iteraciones																
Fase de Puesta en Producción																

Fuente: Autoría propia

2.3.2.2. Presupuesto.

Tabla 3

Presupuesto de la Herramienta

	Materiales	Unidad	Cantidad	Precio (S/.)	Total (S/.)
Materiales	Silla	Unidad	1	50.00	50.00
	Escritorio	Unidad	1	150.00	150.00
	Movilidad y viático	días	144	1.50	216.00
	Lapiceros	Unidad	3	1.00	3.00
	Lápiz	Unidad	1	1.00	1.00
	Borrador	Unidad	1	1.00	1.00
	Tajador	Unidad	1	1.00	1.00
	Hojas bond A4	Unidad	1/2 Millar	15.00	15.00
	Folder o anillado	Unidad	2	5.00	10.00
	Impresiones	Hojas	100	0.10	10.00
	Material bibliográfico virtual	Unidad	10	0.00	0.00
Hardware	Material bibliográfico físico	Unidad	1	30.00	30.00
	Laptop HP	Unidad	1	300.00	300.00
	Impresora	Unidad	1	200.00	200.00
	Memoria USB 100 GB	Unidad	1	100.00	100.00
	Microsoft Office 2013	CD	1	7.00	7.00
Software	Adobe Reader 9	CD	1	3.00	3.00
	Recortes	Unidad	1	0.00	0.00
	Gmail-Drive	-	1	0.00	0.00
	Sistema Operativo Windows 8	Licencia	1	100.00	100.00
	IBM API CONNECT V5 .0	-	1	0.00	0.00
Servicios	Servicio de Internet	Paquete	1	85.00	85.00
	Servicios básico	Paquete	1	50.00	50.00.
					S/.1282.00

Fuente: Autoría propia

2.3.3. Desarrollo de la implementación de la Plataforma API Connect

2.3.3.1. Modelo de la Arquitectura de Integración del Banco Interbank.

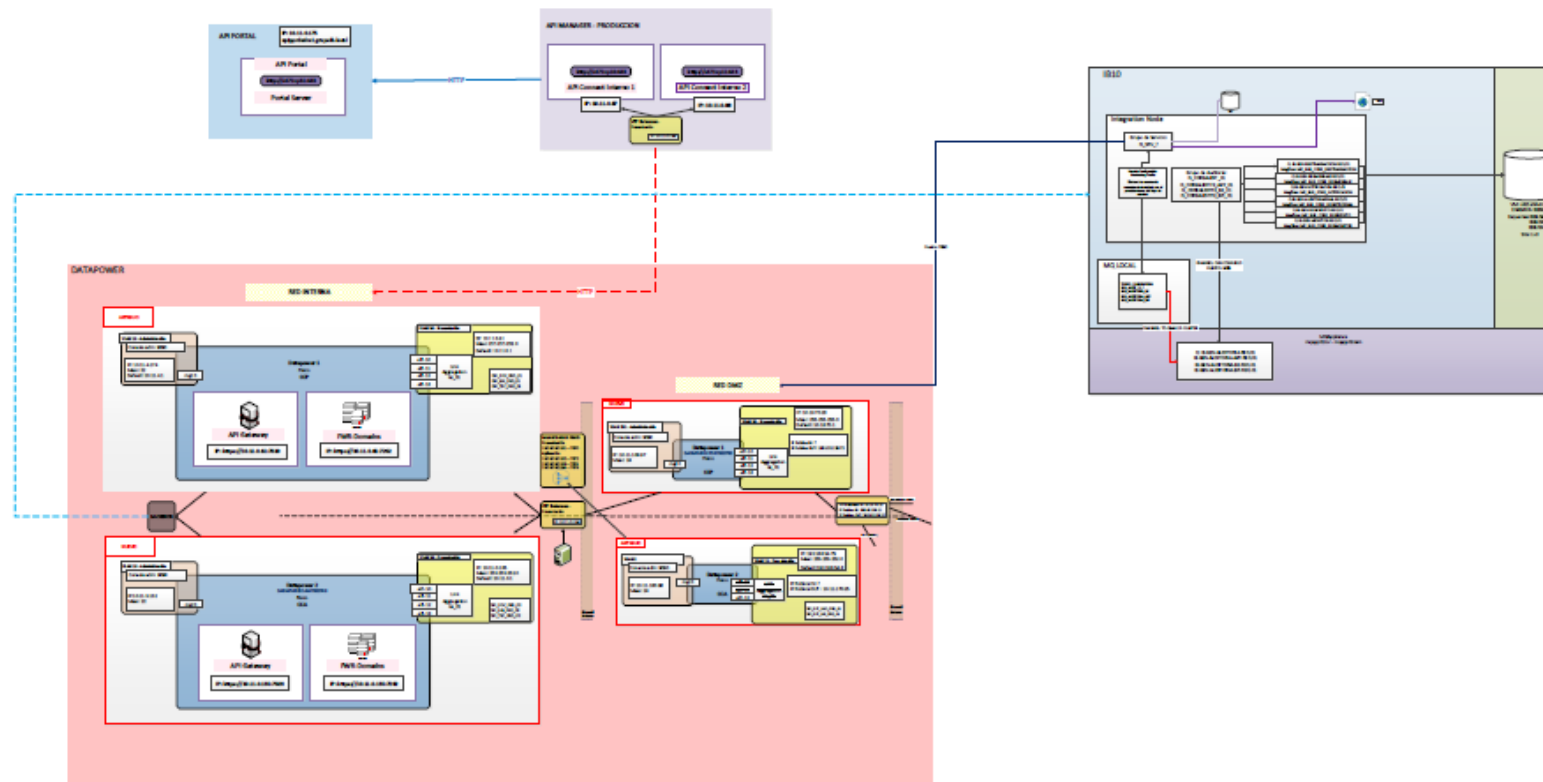


Figura 26. Arquitectura de Integración del Banco Interbank (Elaborado por la investigadora).

2.3.3.2. Validación API input/output fieldnames.

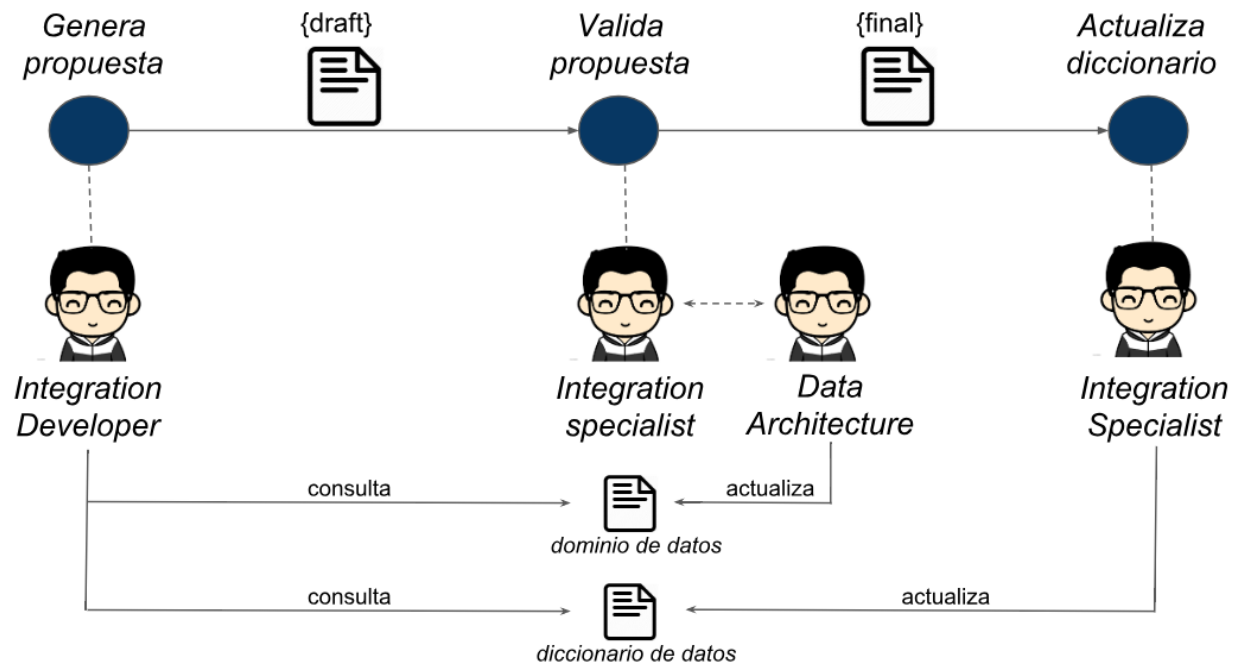


Figura 27. Validación de Integración del Banco Interbank (Elaborado por la investigadora).

2.3.3.3. Contenido de API.

Premisa. Enviar sólo los datos requeridos para ejecutar el servicio.

2.3.3.3.1. Datos.

Los datos que enviamos en el mensaje se clasifican en dos grandes grupos:

- Datos de conectividad u otros requeridos por capas intermedias y/o proveedor para ejecutar el servicio (p.e. clave de acceso de la aplicación consumidora).
- Datos de negocio requeridos para ejecutar el servicio (p.e. si vas a consultar los datos de un cliente, un dato de negocio es el código único del cliente).

2.3.3.3.2. Mensaje.

El mensaje está conformado por dos secciones:

Header

- Datos enviados en la cabecera del protocolo http.
- En esta sección enviamos los datos de conectividad u otros.
- Los datos que se envían en el header son definidos por el área de Gobierno, en caso de requerir un parámetro que no esté contemplado en la lista solicitar su definición al responsable de Gobierno.

Body

- Datos enviados en el contenido del mensaje.
- En esta sección enviamos los datos de negocio.
- Los datos que se envían en el body están estandarizados.

2.3.3.4. Generaciones de API.

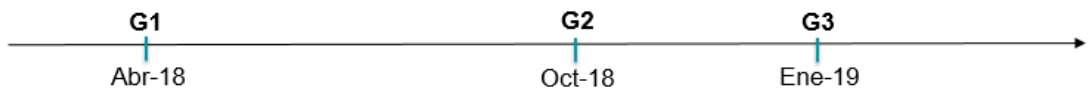


Figura 28. Evolución de las Generaciones de API (Elaborado por la investigadora).

- G1. Orientada a servicios
- G2. Orientada a recursos + http verbs + http header req
- G3. Orientada a recursos + http verbs + input/output estandarizado + ingles

2.3.3.5. HTTP verbs.

Read → GET (provider ws/rest)

Estructura URL: .../API/customer/v1/11223344/basic-info

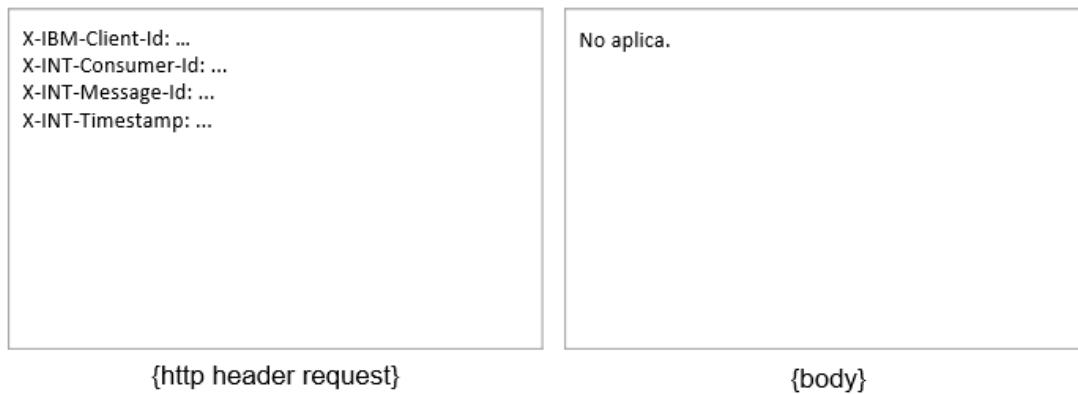


Figura 29. Estructura GET (Elaborado por la investigadora).

Create → POST (provider ws/rest)

Estructura URL: .../API/customer/v1/



Figura 30. Estructura POST (Elaborado por la investigadora).

Update → PUT (provider ws/rest)

Estructura URL: .../API/customer/v1/11223344/basic-info



Figura 31. Estructura PUT (Elaborado por la investigadora).

Delete → DELETE (provider ws/rest)

Estructura URL: .../API/customer/v1/11223344



Figura 32. Estructura DELETE (Elaborado por la investigadora).

2.3.3.6. Códigos de estado de respuesta HTTP.

Respuestas	
200	200 OK
400	400 Bad Request
401	401 Unauthorized
403	403 Error de Negocio
404	404 Not Found
429	429 Too Many Requests
500	500 Internal Server Error

Figura 33. Códigos de respuesta (Elaborado por la investigadora).

- Transacción exitosa

<pre>HTTP/1.1 200 OK X-INT-Messageld-Res: "...."</pre>	<pre>{ "fullName": "Christian Larrauri Albuja", "gender": "M", "maritalStatus": "comprometido", "birthdate": "14-07-1987", "segment": "cliente gold" }</pre>
{http header response}	{body}

Figura 34. Estructura Response exitoso (Elaborado por la investigadora).

- Transacción con errores de estructura



Figura 35. Estructura Response error (Elaborado por la investigadora).

- Transacción no autorizada



Figura 36. Estructura Response no autorizada (Elaborado por la investigadora).

- Errores de negocio



Figura 37. Estructura Response error de negocio (Elaborado por la investigadora).

- Recurso no encontrado



Figura 38. Estructura Response recurso no encontrado (Elaborado por la investigadora).

- Demasiadas transacciones



Figura 39. Estructura Response error exceso de solicitudes (Elaborado por la investigadora).

- Error de servidor: genérico

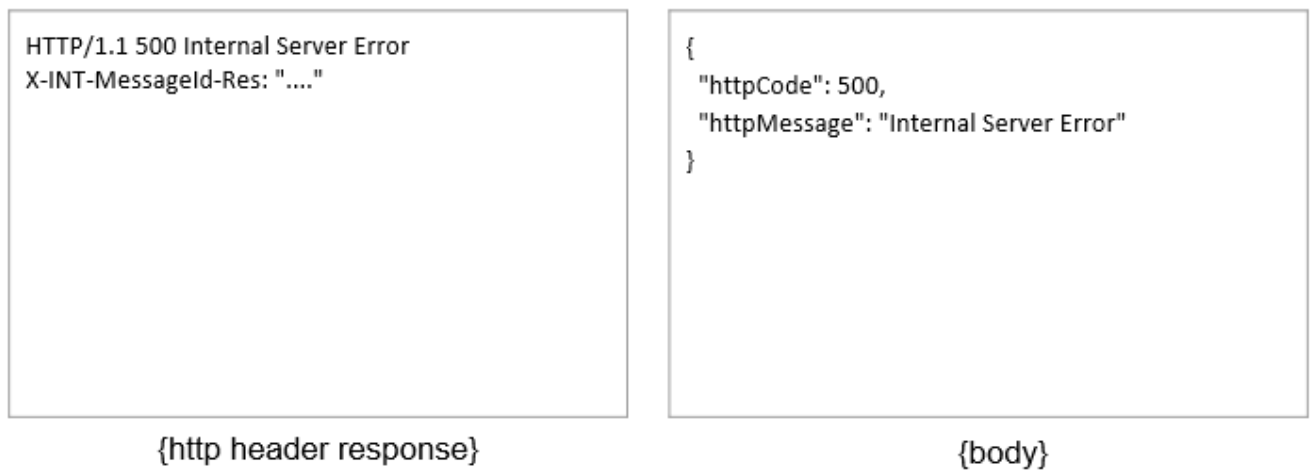


Figura 40. Estructura Response error de servidor (Elaborado por la investigadora).

2.3.4. Desarrollo de la Implementación de la Plataforma API

Connect evaluada

2.3.4.1. Definición de los Requerimientos.

2.3.4.1.1. Servicio Proveedor Watson.

- El servicio proveedor WATSON permite el envío de una notificación a través de un mensaje de texto (SMS).
- La cobertura de envío de SMS a los operadores disponibles de Perú: Movistar, Claro, Entel y Bitel.
- El formato para el número celular debe llevar antepuesto el código postal '51' correspondiente a Perú, por ejemplo: '51999888777'
- El servicio permite el envío individual de mensajes de texto.
- El servicio permite el envío masivo de mensajes de texto.
- El servicio permite personalizar el contenido del mensaje a enviar.
- El servicio permite personalizar el nombre del destinatario del SMS.

Elaboración de Servicio SOAP

- Se construirá el servicio SOAP a través de la herramienta Integration BUS.
- Se utilizará el lenguaje de programación ESQL.

- Será consumida a través de un archivo WSDL con su respectivo XSD en el formato XML,
- El WSDL servirá como firma de consumo de dicho servicio.

Elaboración de Servicio REST(API)

- Se construirá el servicio REST a través de la herramienta API Connect.
- Se desarrollará mediante un archivo de formato YAML,
- El servicio podrá ser consumido a través de una URL REST.

2.3.4.2. Construcción del Servicio SOAP (IBM Integration BUS).

La arquitectura para la construcción del servicio es la siguiente:

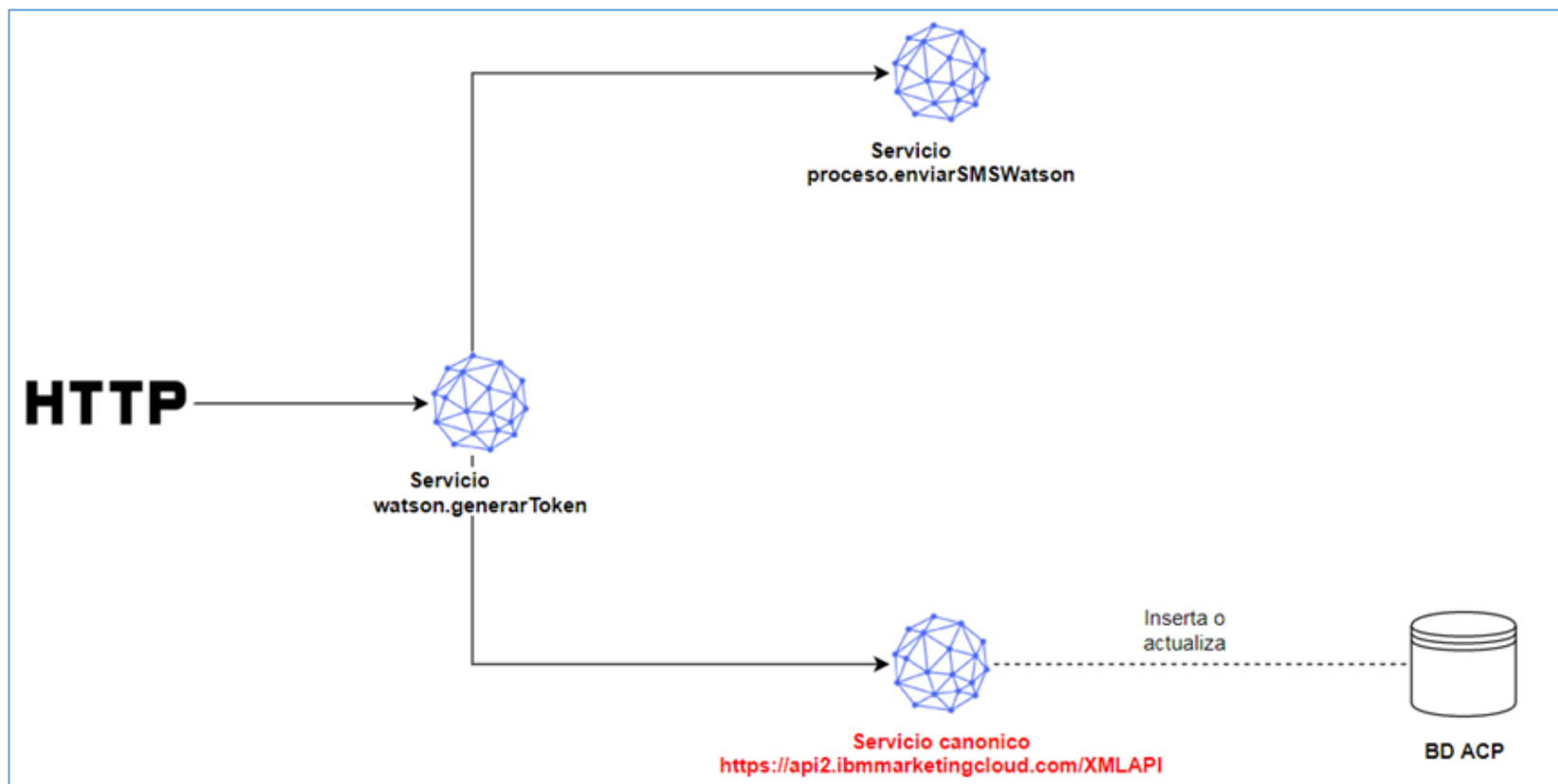


Figura 41. Arquitectura de servicio SOAP (Elaborado por la investigadora).

2.3.4.2.1. Arquitectura y Diseño de la construcción del servicio Canónico Generar Token.

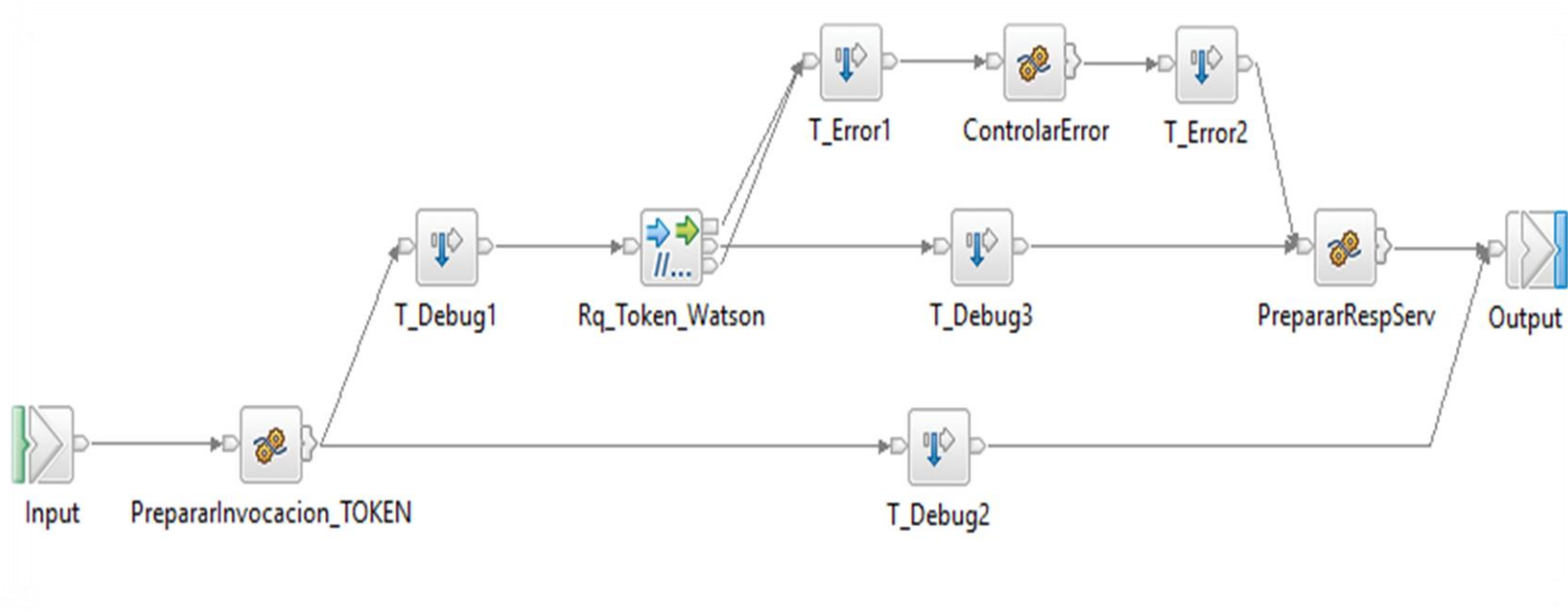


Figura 42. Diseño del servicio SOAP – Canónico (Elaborado por la investigadora).

2.3.4.2.2. Arquitectura y Diseño de la construcción del servicio Compuesto Enviar SMS.

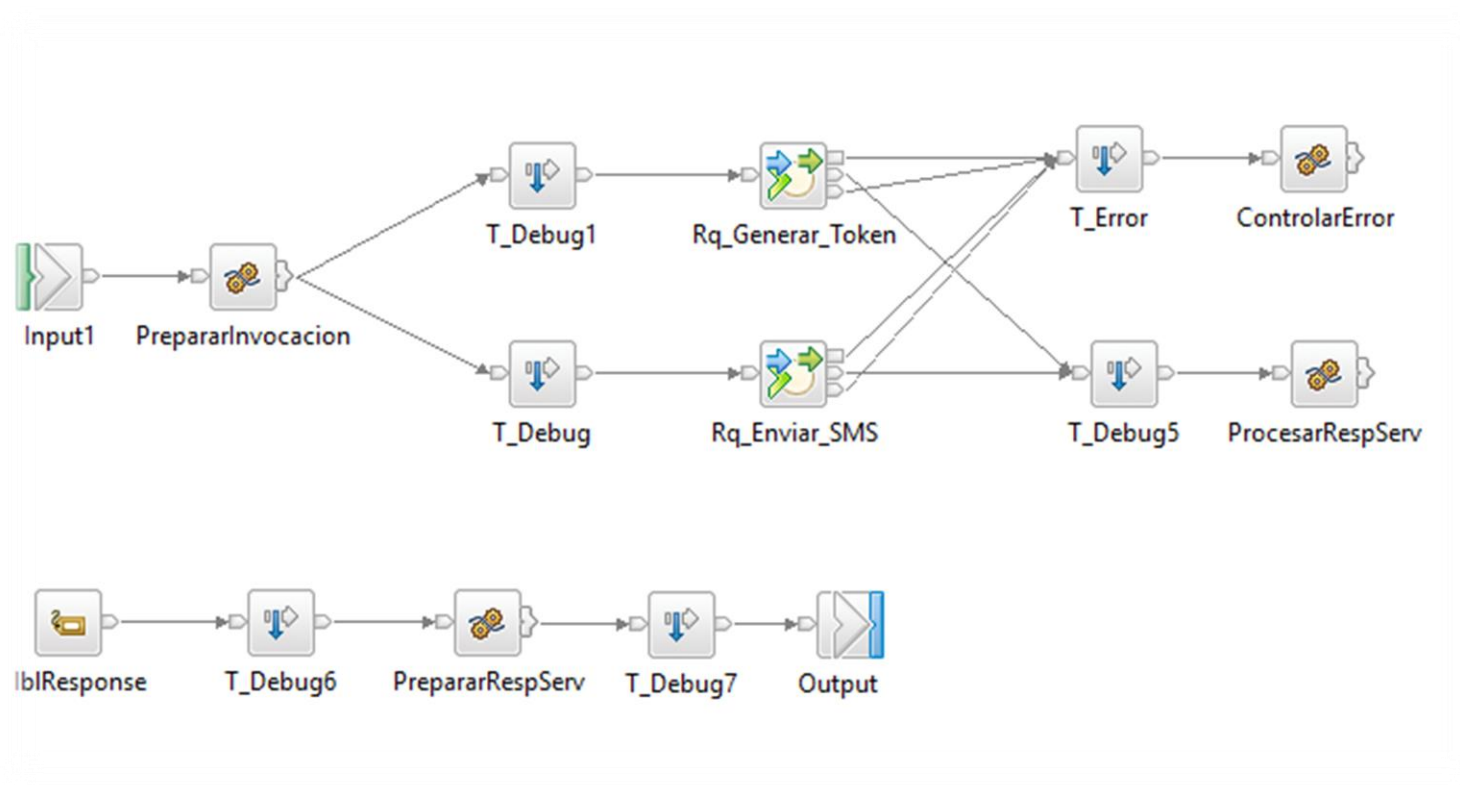


Figura 43. Diseño del servicio SOAP – Compuesto (Elaborado por la investigadora).

2.3.4.2.3. Datos Técnicos del servicio Canónico Generar Token.

Tabla 4

Datos generales del servicio Canónico

		ServicioA0488 - watson.generarToken
DATO		OBJETO
NECESIDAD		Servicio que usa una llave para poder obtener el acceso, esta llave será para generar un token para el consumo de servicios Watson.
URL DEL SERVICIO	DEV	https://dpidev.grupoib.local:3200/ibk/srv/MPO/Utilitarios/watson.generarToken/v1.0
	SIT	https://dpisit.grupoib.local:5200/ibk/srv/MPO/Utilitarios/watson.generarToken/v1.0
	UAT	https://dpiuat.grupoib.local:7200/ibk/srv/MPO/Utilitarios/watson.generarToken/v1.0
GRUPO DE EJECUCIÓN		IS_MPE01_01
VERSIÓN		1

Fuente: Autoría propia

2.3.4.2.4. Datos Técnicos del servicio Compuesto Enviar SMS.

Tabla 5

Datos generales del Servicio Compuesto.

ServicioA0489 - proceso.enviarSMSWatson	
Datos	OBJETO
Necesidad	<p>Armar el uso de dos servicios canónicos para enviar una notificación no transaccional a través de un mensaje de texto</p> <p>Request:</p> <ul style="list-style-type: none"> - Número del teléfono, formato "51927131809". - Una llave para poder obtener el acceso, esta llave será para generar un token para el consumo del servicio.
Url del servicio	<p>DEV https://dpidev.grupoib.local:3200/ibk/srv/MPO/Utilitarios/proceso.enviarSMSWatson/v1.0</p> <p>SIT https://dpisit.grupoib.local:5200/ibk/srv/MPO/Utilitarios/proceso.enviarSMSWatson/v1.0</p> <p>UAT https://dpiuat.grupoib.local:7200/ibk/srv/MPO/Utilitarios/proceso.enviarSMSWatson/v1.0</p>
Grupo de ejecución	IS_MPE01_01
Versión	1

Fuente: Autoría propia

2.3.4.2.5. Descripción de los campos.

Tabla 6

Detalle de campos.

	fieldname	description
<LoginACP>	codigoCliente	Código del cliente
<codigoCliente>d3f0a049</codigoCliente>	codigoSecreto	Código secreto
<codigoSecreto>6c6c4187</codigoSecreto>	actualizarToken	Código de actualización del token
<actualizarToken>rX1Jkrr0ng</actualizarToken>		
</LoginACP>		
<datosMensaje>	contenidoMensaje	Contenido del mensaje
<contenidoMensaje>Hello. Here		
is</contenidoMensaje>	numeroCelular	Número de Teléfono
<contactos>		- Tener en cuenta que empieza con 51
<contacto>	nombreCliente	Nombre del destinatario
<numeroCelular>51927131809</numeroCelular>		
<personalizacion>		
<nombreCliente>Karla		
Vicente</nombreCliente>	identificadorPrograma	Identificador del programa
</personalizacion>		
</contacto>		
</contactos>		
<identificadorPrograma>166150</identificadorPrograma>		
</datosMensaje>		

Fuente: *Autoría propia*

2.3.4.2.6. Ejecución de Servicio SOAP.

Se ejecutó el servicio BUS para envíos de mensajes SMS a través del proveedor Watson, para la prueba se utilizó la herramienta SOAP UI, a continuación, se detallan el Request ejecutado, la respuesta obtenida y la evidencia de que el SMS se envió al celular de destino configurado:

Request. El request utilizado tiene como parámetros de entrada:

- Código de Cliente
- Código Secreto
- Token de Actualización
- Contenido de Mensaje.
- Número de celular
- Nombre cliente
- Identificador del Programa



Figura 44. Request del servicio SOAP (Elaborado por la investigadora).

Response. Como respuesta el servicio presenta los siguientes campos:

- BusResponseMessage
- Locacion
- Identificador

```

<?xml version='1.0' encoding='utf-8'>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:v4="http://intec">
  <soapenv:Header>
    <v4:HeaderResponse>
      <timestamp>2019-11-20T11:00:10.424891-05:00</timestamp>
      <status>
        <responseType>0</responseType>
        <busResponseCode>0</busResponseCode>
        <busResponseMessage>EJECUCION CON EXITO</busResponseMessage>
        <srvResponseCode/>
        <srvResponseMessage/>
      </status>
    </v4:HeaderResponse>
  </soapenv:Header>
  <soapenv:Body>
    <v1:enviarSMSWatsonResponse>
      <datos>
        <locacion>https://api2.ibmmarketingcloud.com/rest/channels/sms/externalconsentsends/62a6c11-16e898b3fa4-1973771dea71da7e4c551ed9f05528be/status</locacion>
        <identificador>62a6c11-16e898b3fa4-1973771dea71da7e4c551ed9f05528be</identificador>
      </datos>
    </v1:enviarSMSWatsonResponse>
  </soapenv:Body>
</soapenv:Envelope>

```

Figura 45. Response del servicio SOAP (Elaborado por la investigadora).

Asimismo, los datos técnicos de la ejecución que nos presenta la herramienta son los siguientes:

Tabla 7

Detalles técnicos de BUS

Cod. Respuesta	HTTP/1.1 200 OK
Server:	Apache-Coyote/1.1
Cod Respuesta Bus	407
Content-Type:	text/xml; charset=utf-8
Fecha hora inicial:	2019-11-20 11:00:09.824479
Fecha hora final:	2019-11-20 11:00:10.425238
Response Time:	865 ms
Transfer:	935 bytes

Fuente: Autoría propia

2.3.4.3. Construcción del Servicio REST (IBM API CONNECT V.5.0).

La arquitectura para la construcción del servicio es la siguiente:

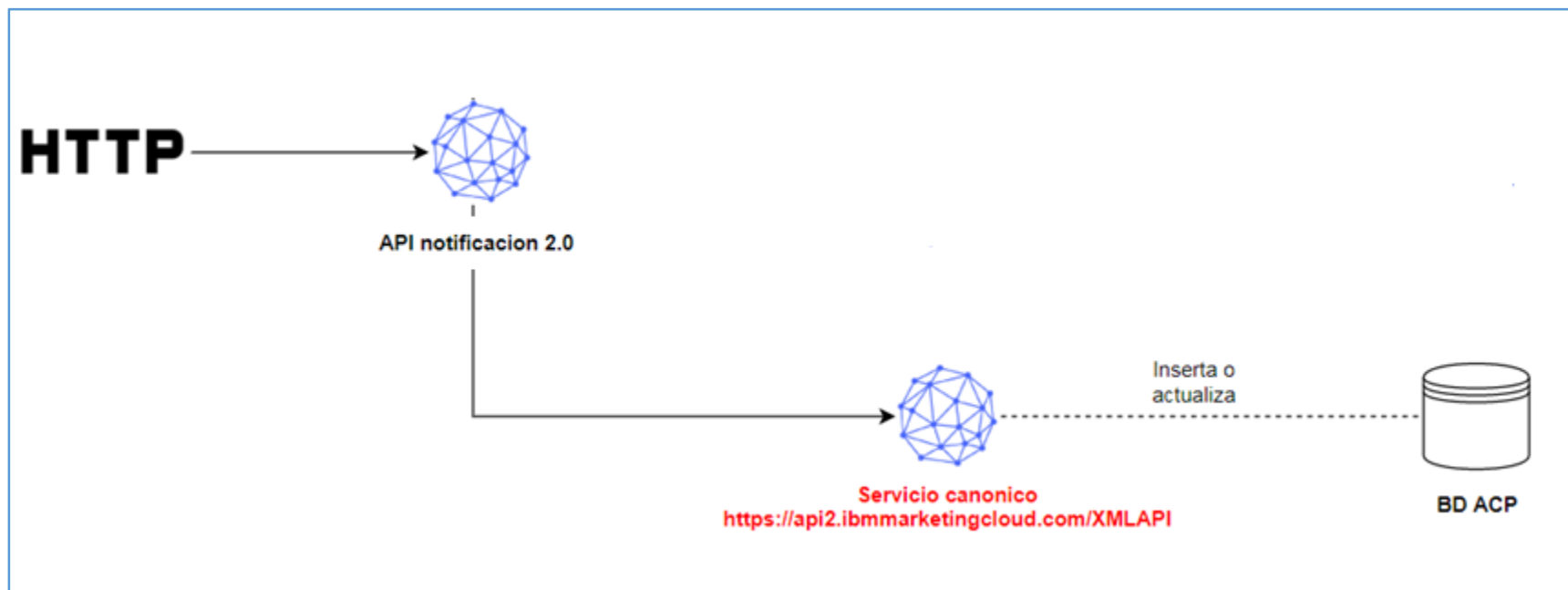


Figura 46. Arquitectura de servicio REST (Elaborado por la investigadora).

2.3.4.3.1. Arquitectura y Diseño de la construcción del servicio REST.

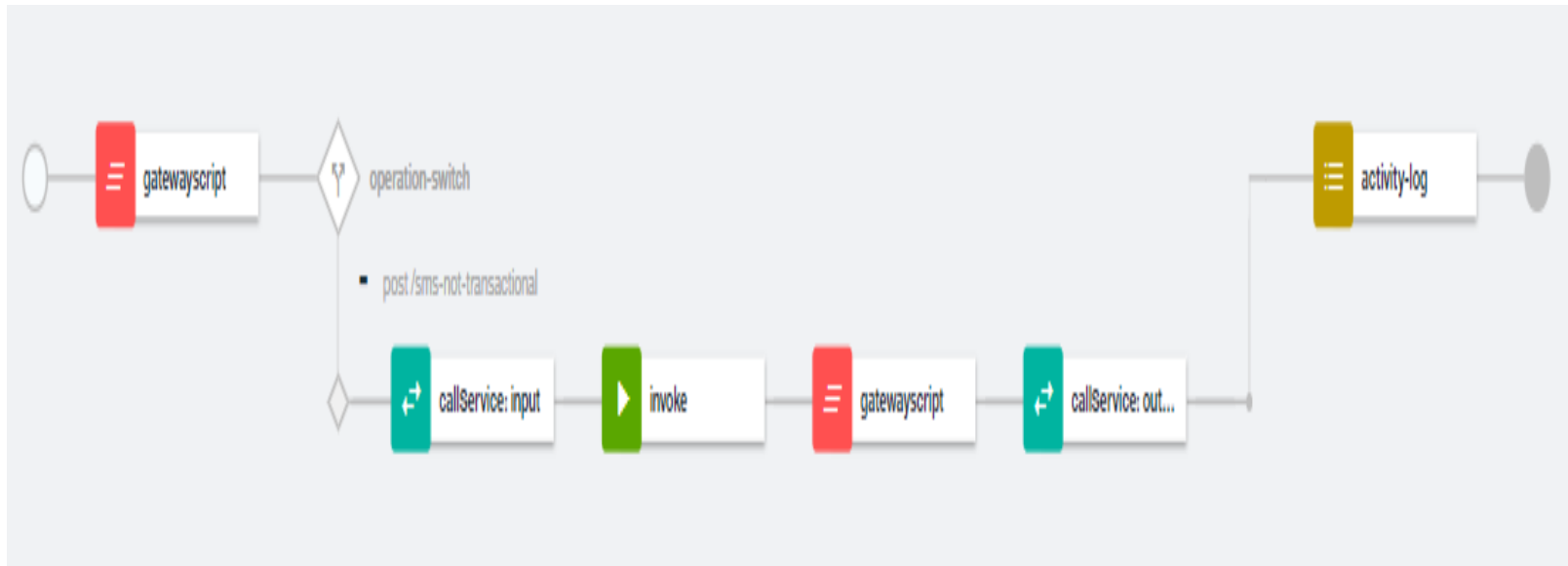


Figura 47. Diseño del servicio API (Elaborado por la investigadora).

2.3.4.3.2. Datos Técnicos del servicio Canónico Generar Token.

Tabla 8

Detalle de datos técnicos.

API notification 2.0	
DATO	OBJETO
Necesidad	<p>Enviar una notificación no transaccional a través de un mensaje de texto</p> <p>Request:</p> <ul style="list-style-type: none"> • Número de teléfono, formato "51927131809". • Una llave para poder obtener el acceso, esta llave será para generar un token para el consumo del servicio.
Url del servicio	<p>DEV https://10.11.40.25:5020/ibk/dev/API/notification/v2/sms-not-transaccional</p> <p>SIT https://10.11.40.25:5020/ibk/sit/API/notification/v2/sms-not-transaccional</p> <p>UAT https://10.11.32.73:7020/ibk/uat/API/notification/v2/sms-not-transaccional</p>
Producto	utilitario
API	notification
Operación	/sms-not-transaccional
Versión	2.0.0

Fuente: Autoría propia

2.3.4.3.3. Descripción de los campos.

Tabla 9

Detalle de datos de los campos

	fieldname	campos en español	descripción
"Login": { "costumerId": "", "clientSecretId": "", "refreshToken": "" }	costumerId	codigoCliente	Código del cliente
	clientSecretId	codigoSecreto	Código secreto
	refreshToken	actualizarToken	Código de actualización del token
"message": { "content": "Hello. Here is the you requested.", "contacts": [{ "cellphone": "51927131809", "name": "Karla Vicente" }], "programId": "166150" }	content	contenidoMensaje	Contenido del mensaje
	cellphone	numeroCelular	Número de Teléfono - Tener en cuenta que empieza con 51
	name	nombreCliente	Nombre del destinatario
	programId	identificadorPrograma	Identificador del programa

Fuente: Autoría propia

JSON REQUEST

```
{
  "Login": {
    "costumerId": "",
    "clientSecretId": "",
    "refreshToken": ""
  },
  "message": {
    "content": "Hello. Here is the you requested.",
    "contacts": [
      {
        "cellphone": "51927131809",
        "name": "Karla Vicente"
      }
    ],
    "programId": "166150"
  }
}
```

Figura 48. JSON Request para consumo del servicio API (Elaborado por la investigadora)

2.3.4.3.4. Ejecución de Servicio REST.

Se ejecutó el servicio API para envíos de mensajes SMS a través del proveedor Watson, para la prueba se utilizó la herramienta SOAP UI, a continuación, se detallan el Request ejecutado, la respuesta obtenida y la evidencia de que el SMS se envió al celular de destino configurado:

Request. El request utilizado tiene como parámetros de entrada:

- costumerId
- clientSecretId
- refreshToken
- content

- cellphone
- name
- programId

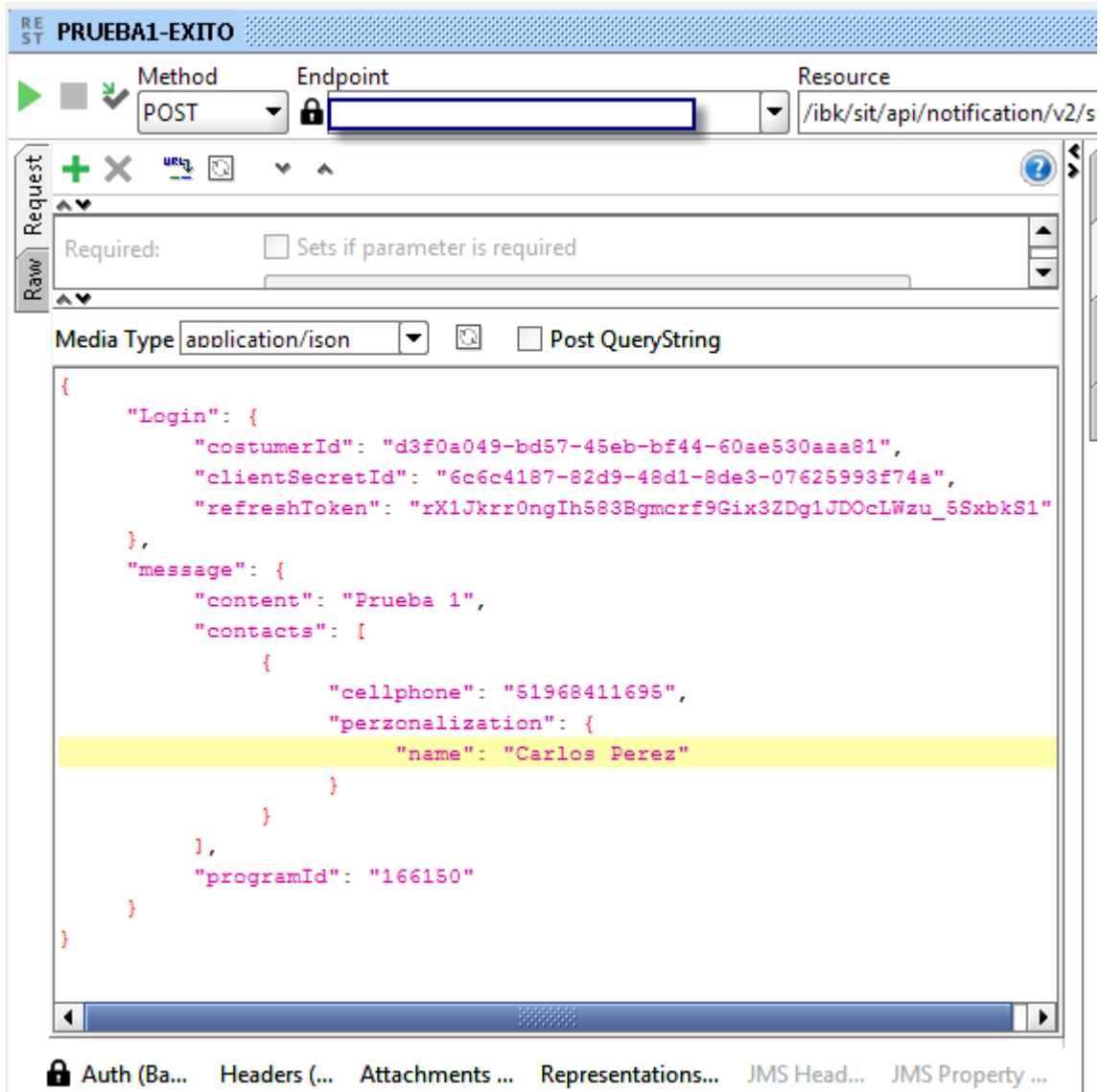


Figura 49. Request del servicio REST (Elaborado por la investigadora).

Response. Como respuesta el servicio presenta los siguientes campos:

- Location
- id

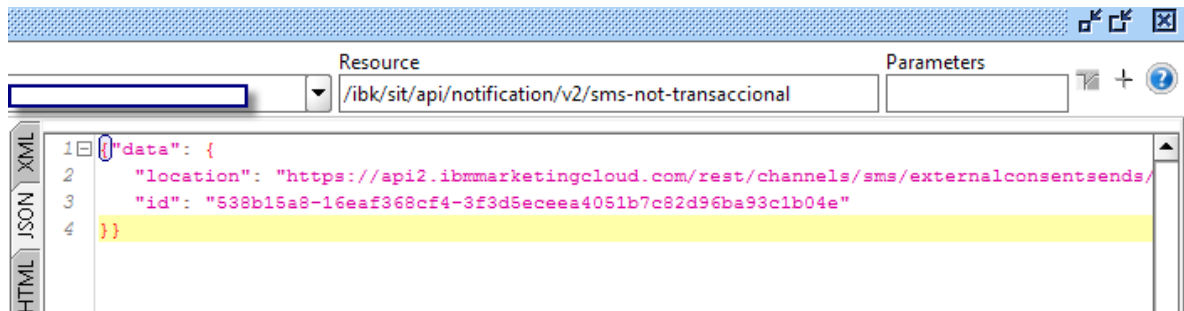


Figura 50. Response del servicio REST (Elaborado por la investigadora).

Asimismo, los datos técnicos de la ejecución que nos presenta la herramienta son los siguientes:

Tabla 10

Detalles técnicos de API

X-Backside-Transport	OK OK
Connection	Keep-Alive
User-Agent	IBM-APIConnect/5.0
X-Global-Transaction-ID	64cd81645ddf07b900ab4553
Accept-Encoding	gzip,deflate
X-Client-IP	10.8.243.98
Date	Wed, 27 Nov 2019 23:33:14 GMT
accept	application/json
Vía	1.1 AQAAAK8PXvE-

Fuente: Autoría propia

2.3.5. Metodología de Evaluación















2.3.5.1. Métodos, técnicas y métricas de recolección de datos.

Evaluación de las métricas e interpretación de Resultados

2.3.5.1.1. Alcance.

Tabla 11

Evaluación de la Métrica de Alcance.

Canales consumidores	IBM INTEGRATION BUS	IBM API CONNECT
ATM - Cajeros		
Servicios POS		
BPI - Banca por Internet		
Siebel -Gestión de Clientes		
Banca SMS		
APP Móvil		
Servicios basados en Cloud		

Fuente: Autoría propia

Interpretación de Datos

De la medición de Alcance respecto a los canales consumidores se obtuvo la siguiente información:

- La plataforma IBM INTEGRATION BUS lo pueden consumir 05 canales que representa el 71.4% del total de canales en el banco que requieren los servicios de integración.
- La plataforma IBM API CONNECT lo pueden consumir 07 canales que representa el 100% del total de canales en el banco que requieren los servicios de integración.

2.3.5.1.2. **Calidad.**

Para los criterios de evaluación que serán consideradas en estas métricas, se les asignó un valor de medición el cual se detalla a continuación:

Tabla 12

Valor de medición de la categoría Confiabilidad.

Descripción de Medición	Ponderado
Alta	10
Moderada	5
Baja	0

Fuente: Autoría propia

Tabla 13

Valor de medición de la categoría Funcionalidad.

Descripción de Medición	Ponderado
Cumple	10
Aceptable	7
Mínima	4
Inaceptable	0

Fuente: Autoría propia

Tabla 14

Valor de medición de la categoría Usabilidad.

Descripción de Medición	Ponderado
Muy fácil	10
Fácil	7
Difícil	4
Muy Difícil	0

Fuente: Autoría propia

Tabla 15

Evaluación de la métrica de calidad – Plataforma Integration BUS.

Plataforma	IBM Integration BUS					
Categoría	Experto 01	Experto 02	Experto 03	Experto 04	Experto 05	Promedio
Confiabilidad	10	5	5	5	5	6.0
Funcionalidad	7	10	4	4	4	5.8
Usabilidad	4	4	0	7	4	3.8

Fuente: Autoría propia

Interpretación de Datos

De la medición de Calidad de la plataforma Integration BUS respecto a la puntuación de las categorías por parte de los expertos del banco que interactúan con ella, se obtuvo la siguiente información:

- Promedio de categoría Confiabilidad.:6.0
- Promedio de categoría Funcionalidad.:5.8
- Promedio de categoría Usabilidad.:3.8

Tabla 16

Evaluación de la métrica de calidad – Plataforma Integration API CONNECT.

Plataforma	IBM API CONNECT					
Categoría	Experto 01	Experto 02	Experto 03	Experto 04	Experto 05	Promedio
Confiabilidad	10	10	5	10	5	8.0
Funcionalidad	10	10	7	10	7	8.8
Usabilidad	7	10	7	7	7	7.6

Fuente: Autoría propia

Interpretación de Datos

De la medición de Calidad de la plataforma Integration API CONNECT respecto a la puntuación de las categorías por parte de los expertos del banco que interactúan con ella, se obtuvo la siguiente información:

- Promedio de categoría Confiabilidad: 8.0
- Promedio de categoría Funcionalidad: 8.8
- Promedio de categoría Usabilidad: 7.6

Tabla 17

Evaluación de la métrica de calidad – Plataforma Integration BUS vs API Connect.

Categoría	IBM Integration BUS	IBM API CONNECT
Confiabilidad	6.0	8.0
Funcionalidad	5.8	8.8
Usabilidad	3.8	7.6

Fuente: Autoría propia

Interpretación de Datos

El promedio del puntaje de Calidad de la plataforma de IBM Integration BUS es de 5.2 mientras que la medición de Calidad de la plataforma API Connect nos muestra un resultado promedio de 8.1.

2.3.5.1.3. *Tiempo de respuesta.*

Para medir los tiempos de respuesta primero se efectuó una prueba de carga al servicio y luego una prueba de concurrencia como se evidencia en la documentación siguiente:

Ejecución de Pruebas de Carga al Servicio SOAP

Prueba de Carga. Se ejecutó una prueba de carga del servicio BUS para envíos de mensajes SMS a través del proveedor Watson, para la prueba se utilizó la JMETER versión 5.0, a continuación, se detallan los resultados de la prueba:

- *Archivos de configuración*

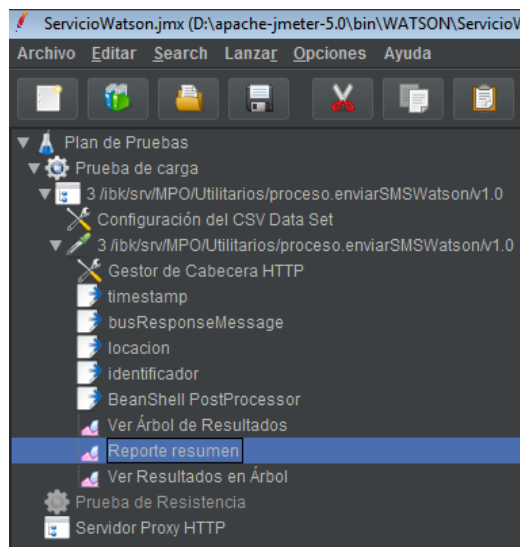


Figura 51. Archivo de configuración del script de Carga (Elaborado por la investigadora).

- *Parametrización de los campos de entrada.* Se parametrizan los campos de entrada y se configura el datapool con formato *.csv (valores separados por coma) para el cargado automático de los datos.

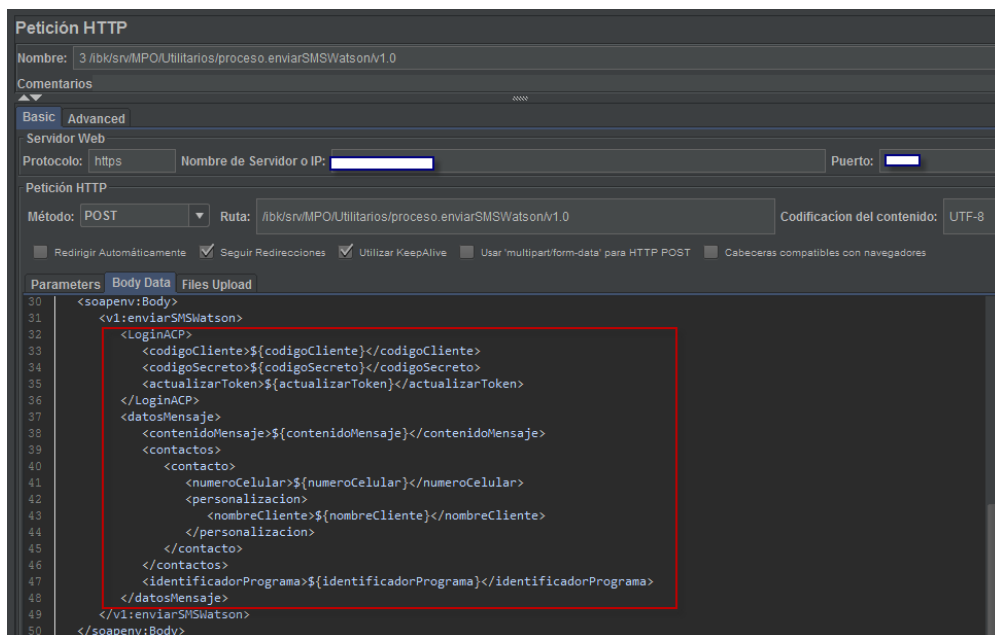


Figura 52. Parametrización de los campos de entrada (Elaborado por la investigadora).

Se utiliza el componente “configuración del CSV Data Set” para hacer un llamado al archivo datapool.csv con los parámetros de entrada de la petición.

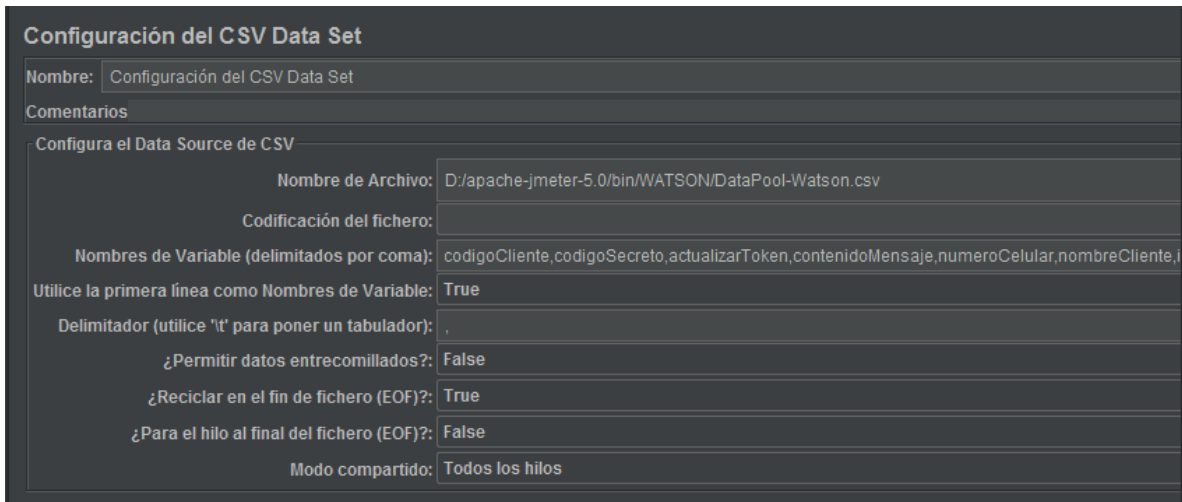


Figura 53. Configuración del Data SET (Elaborado por la investigadora).

El Archivo Datapool.csv donde se han cargado los números de celulares para la prueba es el siguiente

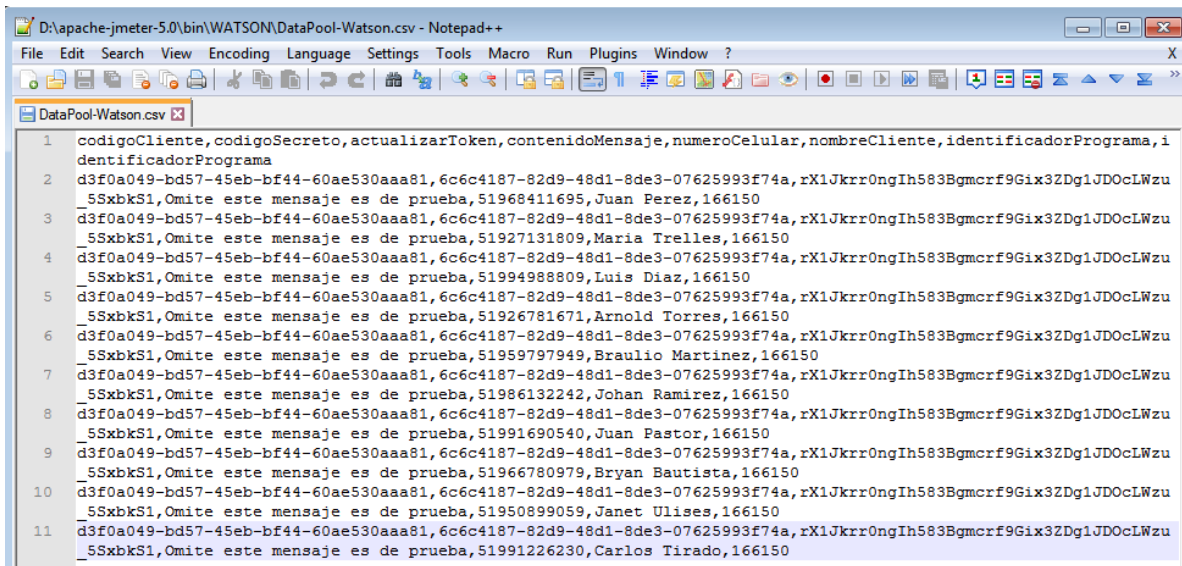


Figura 54. Archivo Datapool.csv (Elaborado por la investigadora).

Reporte del resumen de la prueba

Se realizó la ejecución de carga con 10 hilos

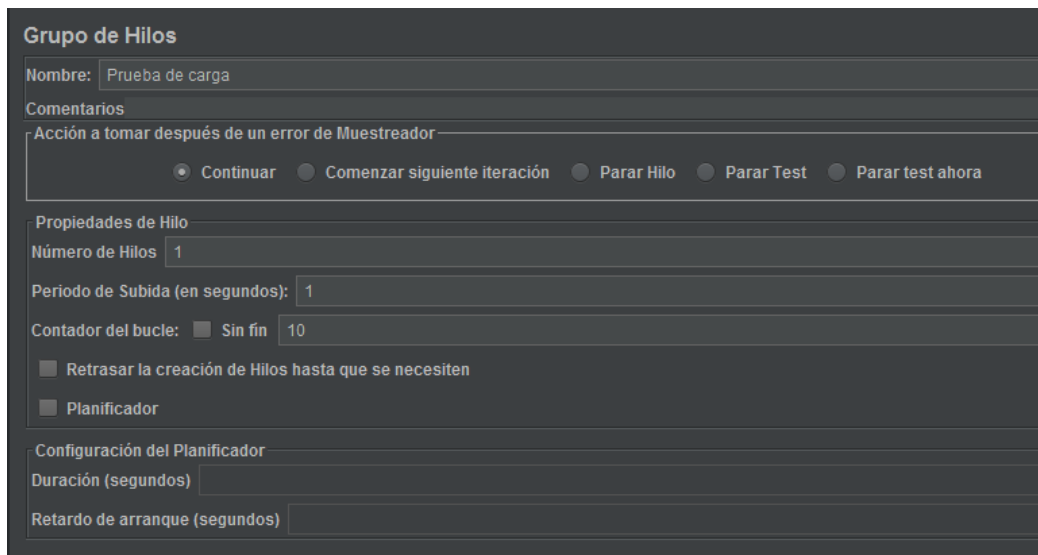


Figura 55. Configuración del grupo de hilos para la ejecución de carga (Elaborado por la investigadora).

Se verifica que las ejecuciones de la prueba hayan sido exitosas:

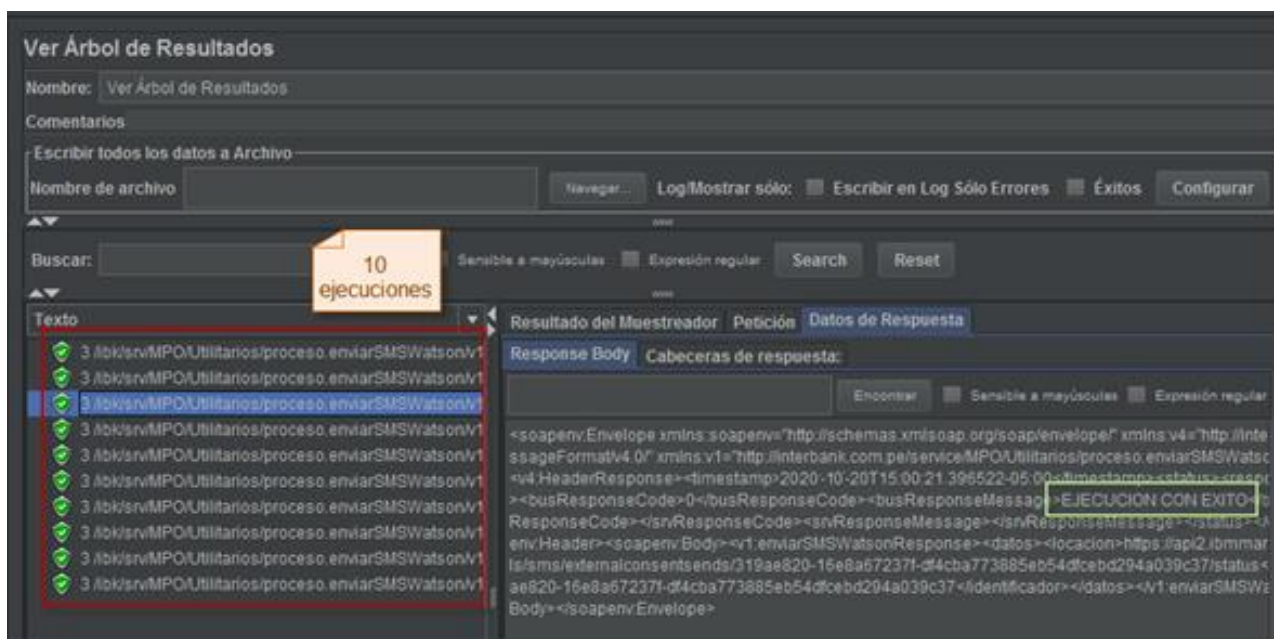


Figura 56. Lista de las ejecuciones realizadas (Elaborado por la investigadora).

Los tiempos de Muestra de cada una de las ejecuciones se detallan en la siguiente imagen

Ver Resultados en Árbol

Nombre: Ver Resultados en Árbol

Comentarios

Escribir todos los datos a Archivo

Nombre de archivo Navegar... Log/Mostrar sólo: Escribir en Log Sólo Errores Éxitos

Muestra #	Tiempo de co...	Nombre del hilo	Etiqueta	Tiempo de Mue...	Estado	Bytes	Sent Bytes	Latency	Connect Time(...)
1	15:00:18.527	Prueba de car...	3 /ibk/srv/MPO/...	2419	✓	855	2529	2418	31
2	15:00:20.954	Prueba de car...	3 /ibk/srv/MPO/...	691	✓	855	2532	691	42
3	15:00:21.648	Prueba de car...	3 /ibk/srv/MPO/...	818	✓	855	2528	817	34
4	15:00:22.469	Prueba de car...	3 /ibk/srv/MPO/...	659	✓	855	2532	659	36
5	15:00:23.132	Prueba de car...	3 /ibk/srv/MPO/...	395	✓	856	2535	395	35
6	15:00:23.529	Prueba de car...	3 /ibk/srv/MPO/...	280	✓	855	2532	280	26
7	15:00:23.810	Prueba de car...	3 /ibk/srv/MPO/...	585	✓	855	2530	585	27
8	15:00:24.398	Prueba de car...	3 /ibk/srv/MPO/...	233	✓	855	2533	233	39
9	15:00:24.635	Prueba de car...	3 /ibk/srv/MPO/...	343	✓	855	2531	343	33
10	15:00:24.981	Prueba de car...	3 /ibk/srv/MPO/...	500	✓	855	2532	500	50

Figura 57. Lista de los tiempos de muestra de las ejecuciones realizadas (Elaborado por la investigadora).

Reporte resumen

Nombre: Reporte resumen

Comentarios

Escribir todos los datos a Archivo

Nombre de archivo Navegar... Log/Mostrar sólo: Escribir en Log Sólo Errores Éxitos

Etiqueta	# Muestras	Media	Mín	Máx	Desv. Está...	% Error	Rendimiento	Kb/sec	Sent KB/sec	Media de Bytes
3 /ibk/srv/M...	10	692	233	2419	603.18	0.00%	1.4/sec	1.20	3.55	855.1
Total	10	692	233	2419	603.18	0.00%	1.4/sec	1.20	3.55	855.1

Figura 58. Reporte de resumen de la prueba de carga ejecutada (Elaborado por la investigadora).

Tabla 18

Resumen de la prueba de carga al servicio BUS

Etiqueta	Muestras	Media (ms)	Min (ms)	Máx (ms)	Desv. Estandar	%Error	Rendimiento	Media de Bytes
proceso.enviarSMSWatson/v1.0	10	692	233	2419	603.18	0	1.44	855.10
Total	10	692	233	2419	603.18	0	1.44	855.10

Fuente: Autoría propia

Ejecución de Prueba de Concurrencia al Servicio SOAP

Se ejecutó una prueba de resistencia al servicio BUS para envíos de mensajes SMS a través del proveedor Watson, para la prueba se utilizó

la JMETER versión 5.0, a continuación, se detallan los resultados de la prueba:

Archivos de configuración

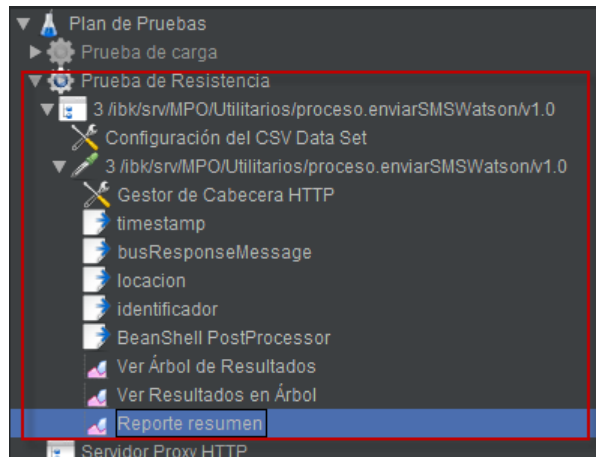


Figura 59. Archivo de configuración del script de Concurrencia (Elaborado por la investigadora).

Se verifica que las ejecuciones la prueba hayan sido exitosas

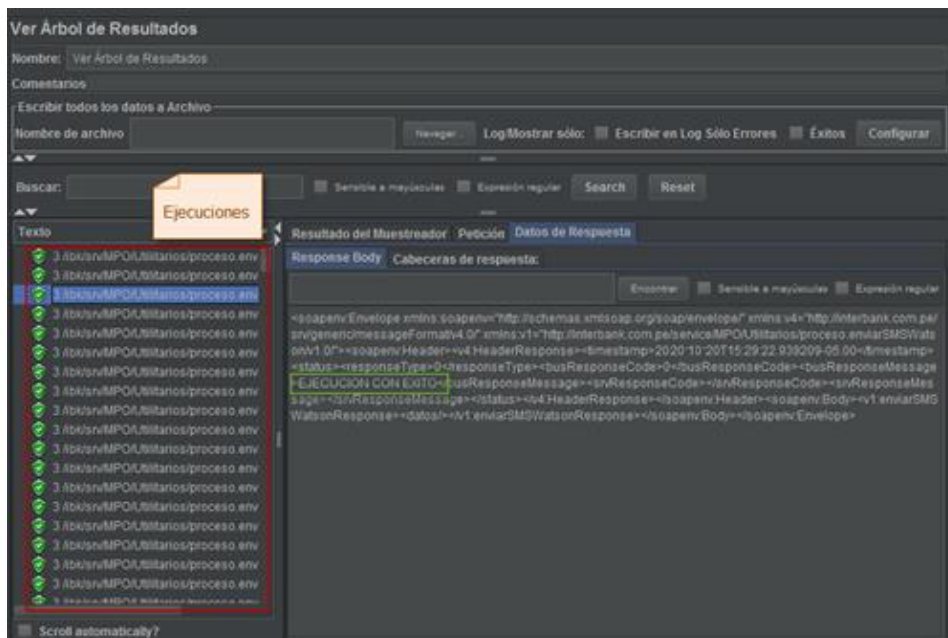


Figura 60. Lista de las ejecuciones realizadas (Elaborado por la investigadora).

Se realizó la ejecución de carga con 05 usuarios concurrentes durante 05 minutos.

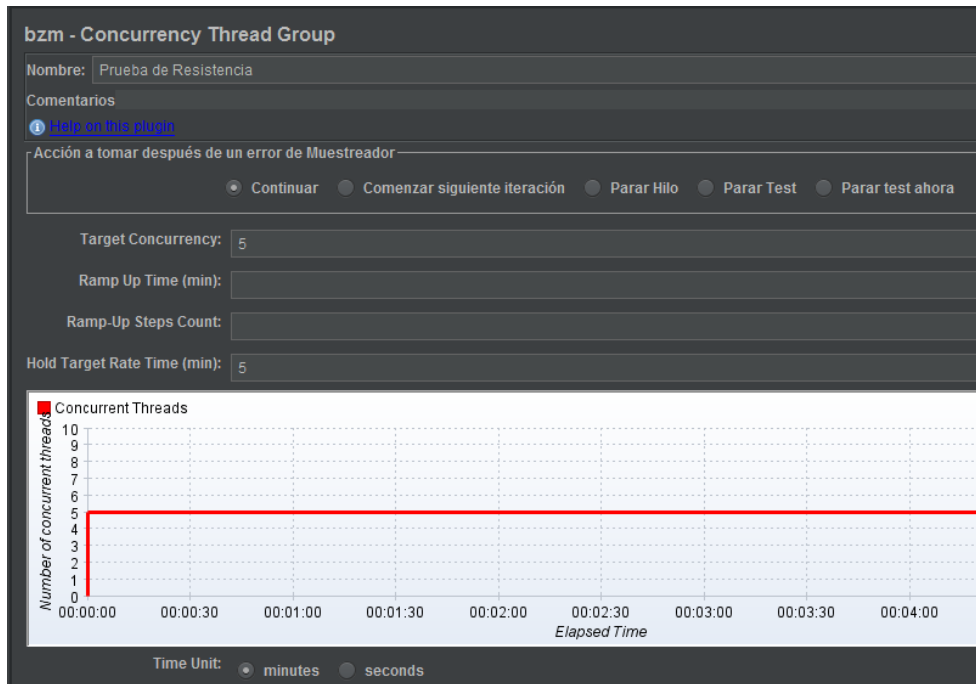


Figura 61. Configuración del grupo de concurrencia para la ejecución de carga (Elaborado para la investigadora).

Los tiempos de Muestra de cada una de las ejecuciones se detallan en la siguiente imagen

Ver Resultados en Árbol

Nombre: Ver Resultados en Árbol

Comentarios

Escribir todos los datos a Archivo

Nombre de archivo: Navegar... Log/Mostrar sólo: Escribir en Log Sólo Errores Éxitos

Muestra #	Tiempo de co...	Nombre del hilo	Etiqueta	Tiempo de Mu...	Estado	Bytes	Sent Bytes	Latency	Connect Time(...)
1	15:26:21.931	Prueba de Re...	3 /ibk/srvMPO/...	711	✓	735	2518	711	59
2	15:26:21.948	Prueba de Re...	3 /ibk/srvMPO/...	696	✓	734	2524	696	39
3	15:26:21.943	Prueba de Re...	3 /ibk/srvMPO/...	712	✓	735	2521	712	57
4	15:26:21.932	Prueba de Re...	3 /ibk/srvMPO/...	766	✓	734	2521	766	42
5	15:26:21.940	Prueba de Re...	3 /ibk/srvMPO/...	955	✓	734	2517	955	331
6	15:26:22.660	Prueba de Re...	3 /ibk/srvMPO/...	567	✓	734	2522	566	38
7	15:26:22.650	Prueba de Re...	3 /ibk/srvMPO/...	580	✓	735	2521	580	32
8	15:26:22.701	Prueba de Re...	3 /ibk/srvMPO/...	533	✓	734	2520	533	28
9	15:26:22.897	Prueba de Re...	3 /ibk/srvMPO/...	448	✓	735	2521	448	27
10	15:26:22.653	Prueba de Re...	3 /ibk/srvMPO/...	760	✓	735	2519	760	49
11	15:26:23.235	Prueba de Re...	3 /ibk/srvMPO/...	519	✓	735	2517	519	112
12	15:26:23.347	Prueba de Re...	3 /ibk/srvMPO/...	428	✓	735	2521	428	24
13	15:26:23.232	Prueba de Re...	3 /ibk/srvMPO/...	624	✓	735	2521	624	113
14	15:26:23.414	Prueba de Re...	3 /ibk/srvMPO/...	563	✓	735	2524	563	76
15	15:26:23.229	Prueba de Re...	3 /ibk/srvMPO/...	827	✓	735	2518	827	108
16	15:26:23.755	Prueba de Re...	3 /ibk/srvMPO/...	459	✓	733	2521	459	45
17	15:26:23.776	Prueba de Re...	3 /ibk/srvMPO/...	508	✓	735	2519	507	29
18	15:26:23.857	Prueba de Re...	3 /ibk/srvMPO/...	515	✓	735	2522	515	52
19	15:26:23.980	Prueba de Re...	3 /ibk/srvMPO/...	462	✓	735	2520	462	49
20	15:26:24.057	Prueba de Re...	3 /ibk/srvMPO/...	501	✓	735	2521	500	49
21	15:26:24.216	Prueba de Re...	3 /ibk/srvMPO/...	528	✓	735	2518	528	43
22	15:26:24.286	Prueba de Re...	3 /ibk/srvMPO/...	498	✓	735	2521	496	65
23	15:26:24.374	Prueba de Re...	3 /ibk/srvMPO/...	442	✓	735	2517	441	60
24	15:26:24.445	Prueba de Re...	3 /ibk/srvMPO/...	406	✓	735	2521	405	32
25	15:26:24.561	Prueba de Re...	3 /ibk/srvMPO/...	479	✓	735	2524	478	38
26	15:26:24.746	Prueba de Re...	3 /ibk/srvMPO/...	439	✓	735	2521	439	38

Scroll automatically? Child samples? No. de Muestras 1937 Última Muestra 13895 Media 701 Desviación 808

Figura 62. Lista de los tiempos de muestra de las ejecuciones realizadas (Elaborado por la investigadora).

Como resultado de la prueba se obtuvo lo siguiente:

Con 05 usuarios concurrentes durante 05 minutos se realizaron 1,937 ejecuciones, con una media de ejecución de 781 milisegundos y con un tiempo mínimos de 330 ms y máximos de 13,895 milisegundos.

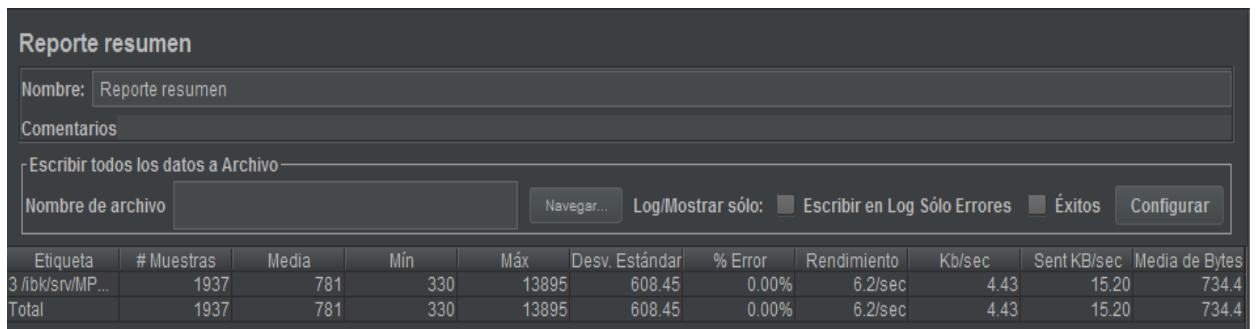


Figura 63. Reporte de resumen de la prueba de concurrencia ejecutada (Elaborado por la investigadora).

Tabla 19

Resumen de la prueba de concurrencia al servicio BUS.

Etiqueta	Muestras	Media (ms)	Min (ms)	Máx (ms)	Desv. Estándar	% Error	Rendimiento	Media de Bytes
proceso.enviarSMSWatson/v1.0	1937	781	330	13895	608.45	0	6.18	734.44
Total	1937	781	330	13895	608.45	0	6.18	734.44

Fuente: Autoría propia

2.3.5.1.4. Ejecución de Pruebas de Carga al Servicio API

Se ejecutó una prueba de carga del servicio API para envíos de mensajes SMS a través del proveedor Watson, para la prueba se utilizó la JMETER versión 5.0, a continuación, se detallan los resultados de la prueba:

Archivos de configuración

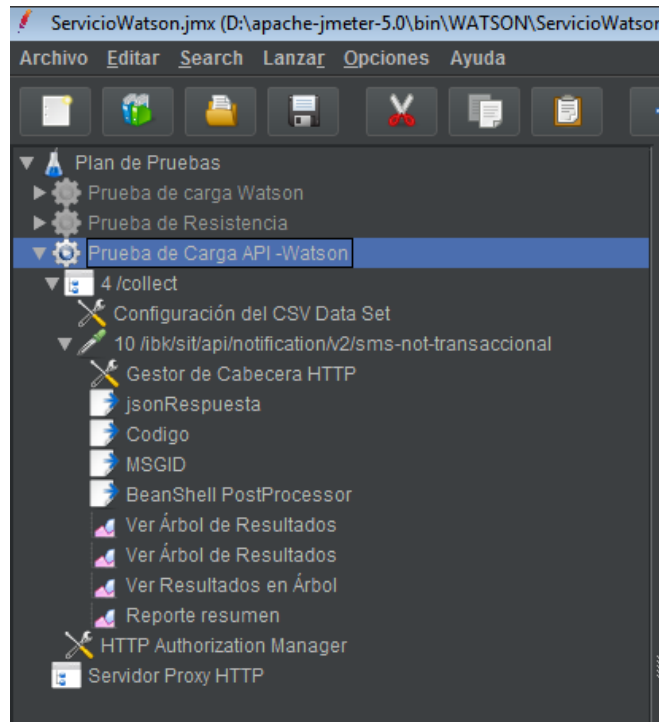


Figura 64. Archivo de configuración del script de Carga (Elaborado por la investigadora).

Parametrización de los campos de entrada. Se parametrizan los campos de entrada y se configura el datapool con formato *.csv (valores separados por coma) para el cargado automático de los datos.

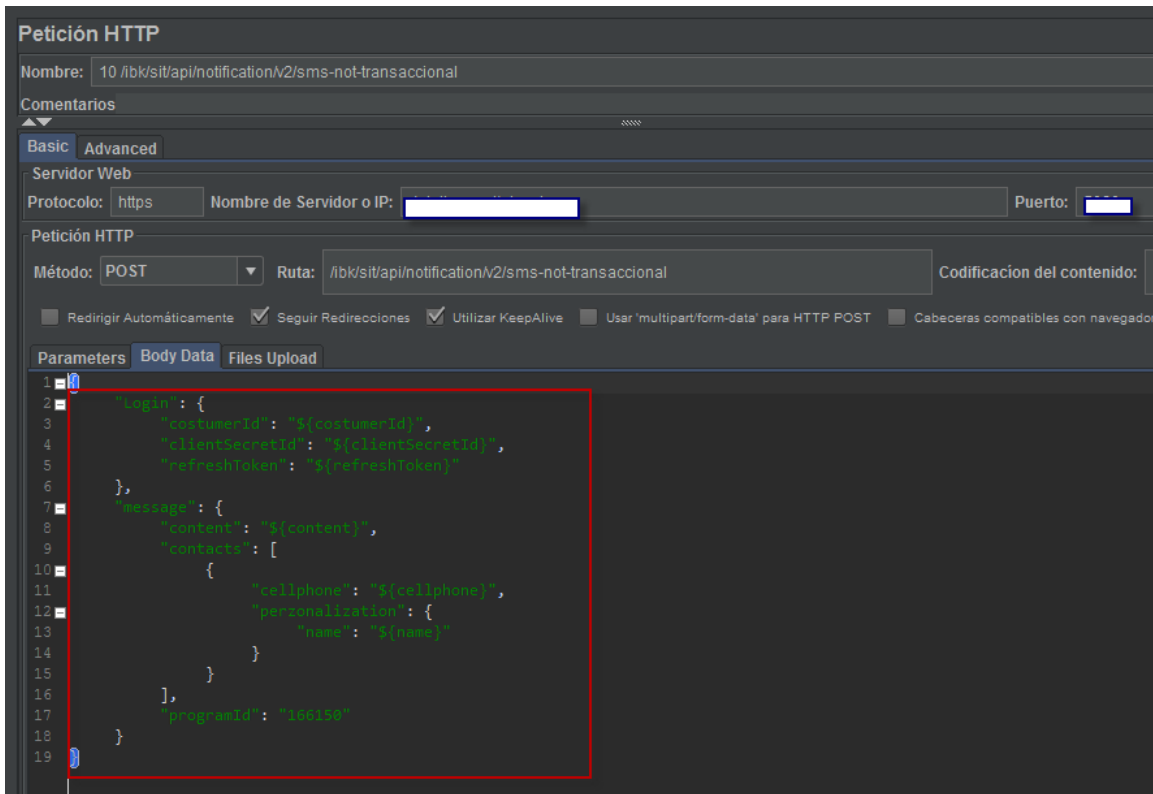


Figura 65. Parametrización de los campos de entrada (Elaborado por la investigadora).

Se utiliza el componente “configuración del CSV Data Set” para hacer un llamado al archivo datapool.csv con los parámetros de entrada de la petición.

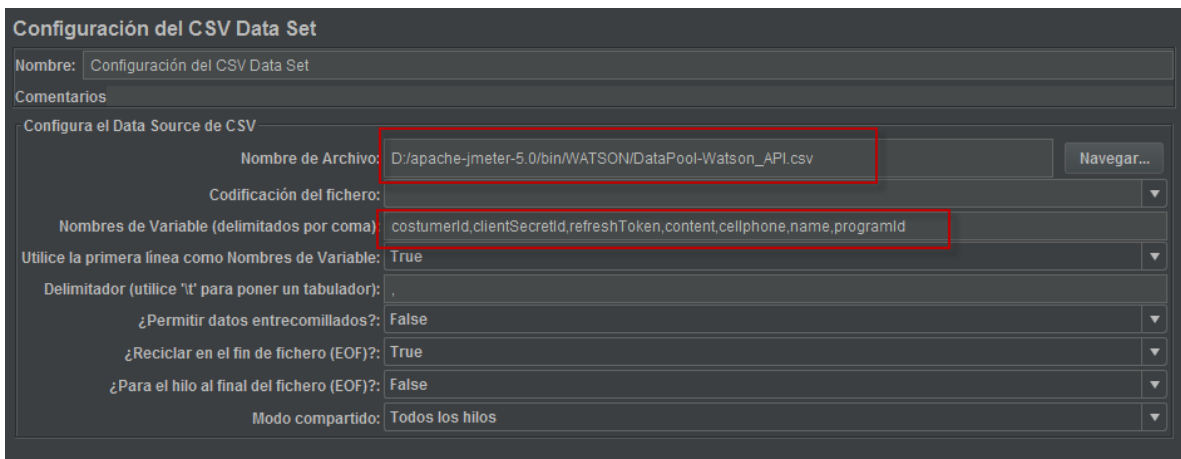


Figura 66. Configuración del Data SET (Elaborado por la investigadora).

Archivo Datapool.csv donde se han cargado los números de celulares para la prueba

```

D:\apache-jmeter-5.0\bin\WATSON\DataPool-Watson_API.csv - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
DataPool-Watson_API.csv
1  customerId,clientSecretId,refreshToken,content,cellphone,name,programId
2  d3f0a049-bd57-45eb-bf44-60ae530aaa81,6c6c4187-82d9-48d1-8de3-07625993f74a,rX1Jkrr0ngIh583Bgmcrf9Gix3ZDg1JD0cLWzu_5SxbkS1,Omite este mensaje es de
prueba,51968411695,Juan Perez,166150
3  d3f0a049-bd57-45eb-bf44-60ae530aaa81,6c6c4187-82d9-48d1-8de3-07625993f74a,rX1Jkrr0ngIh583Bgmcrf9Gix3ZDg1JD0cLWzu_5SxbkS1,Omite este mensaje es de
prueba,51927131809,Maria Trelles,166150
4  d3f0a049-bd57-45eb-bf44-60ae530aaa81,6c6c4187-82d9-48d1-8de3-07625993f74a,rX1Jkrr0ngIh583Bgmcrf9Gix3ZDg1JD0cLWzu_5SxbkS1,Omite este mensaje es de
prueba,51994988809,Luis Diaz,166150
5  d3f0a049-bd57-45eb-bf44-60ae530aaa81,6c6c4187-82d9-48d1-8de3-07625993f74a,rX1Jkrr0ngIh583Bgmcrf9Gix3ZDg1JD0cLWzu_5SxbkS1,Omite este mensaje es de
prueba,51926781671,Arnold Torres,166150
6  d3f0a049-bd57-45eb-bf44-60ae530aaa81,6c6c4187-82d9-48d1-8de3-07625993f74a,rX1Jkrr0ngIh583Bgmcrf9Gix3ZDg1JD0cLWzu_5SxbkS1,Omite este mensaje es de
prueba,51959797949,Braulio Martinez,166150
7  d3f0a049-bd57-45eb-bf44-60ae530aaa81,6c6c4187-82d9-48d1-8de3-07625993f74a,rX1Jkrr0ngIh583Bgmcrf9Gix3ZDg1JD0cLWzu_5SxbkS1,Omite este mensaje es de
prueba,51986132242,Johan Ramirez,166150
8  d3f0a049-bd57-45eb-bf44-60ae530aaa81,6c6c4187-82d9-48d1-8de3-07625993f74a,rX1Jkrr0ngIh583Bgmcrf9Gix3ZDg1JD0cLWzu_5SxbkS1,Omite este mensaje es de
prueba,51991690540,Juan Pastor,166150
9  d3f0a049-bd57-45eb-bf44-60ae530aaa81,6c6c4187-82d9-48d1-8de3-07625993f74a,rX1Jkrr0ngIh583Bgmcrf9Gix3ZDg1JD0cLWzu_5SxbkS1,Omite este mensaje es de
prueba,51966780979,Bryan Bautista,166150
10 d3f0a049-bd57-45eb-bf44-60ae530aaa81,6c6c4187-82d9-48d1-8de3-07625993f74a,rX1Jkrr0ngIh583Bgmcrf9Gix3ZDg1JD0cLWzu_5SxbkS1,Omite este mensaje es de
prueba,51950899059,Janet Ulises,166150
11 d3f0a049-bd57-45eb-bf44-60ae530aaa81,6c6c4187-82d9-48d1-8de3-07625993f74a,rX1Jkrr0ngIh583Bgmcrf9Gix3ZDg1JD0cLWzu_5SxbkS1,Omite este mensaje es de
prueba,51991226230,Carlos Tirado,166150

```

Figura 67. Archivo Datapool.csv (Elaborado por la investigadora).

Reporte del resumen de la prueba

Se realizó la ejecución de carga con 10 hilos

Grupo de Hilos

Nombre: Prueba de carga

Comentarios

Acción a tomar después de un error de Muestreador

Continuar
 Comenzar siguiente iteración
 Parar Hilo
 Parar Test
 Parar test ahora

Propiedades de Hilo

Número de Hilos:

Periodo de Subida (en segundos):

Contador del bucle: Sin fin

Retrasar la creación de Hilos hasta que se necesiten
 Planificador

Configuración del Planificador

Duración (segundos)

Retardo de arranque (segundos)

Figura 68. Configuración del grupo de hilos para la ejecución de carga (Elaborado por la investigadora).

Se verifica que las ejecuciones de la prueba hayan sido exitosas

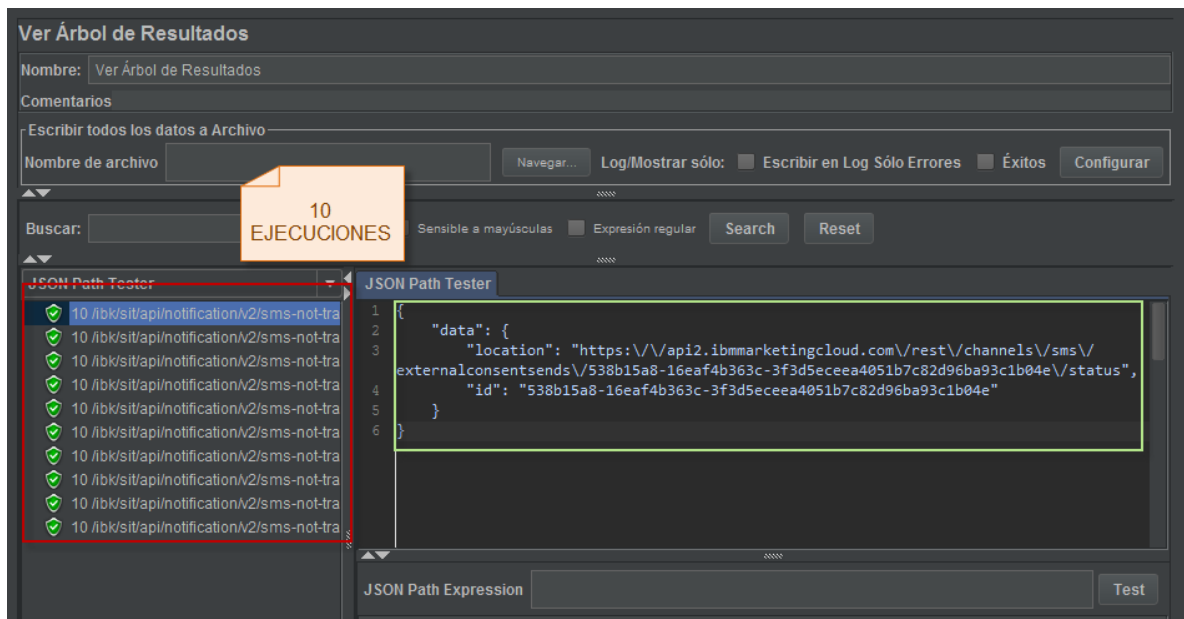


Figura 69. Lista de las ejecuciones realizadas. (Elaborado por la investigadora).

Los tiempos de Muestra de cada una de las ejecuciones se detallan en la siguiente figura.

Ver Resultados en Árbol

Nombre: Ver Resultados en Árbol

Comentarios

Escribir todos los datos a Archivo

Nombre de archivo Navegar... Log/Mostrar sólo: Escribir en Log Sólo Errores Éxitos Configurar

Muestra #	Tiempo de co...	Nombre del hilo	Etiqueta	Tiempo de Mue...	Estado	Bytes	Sent Bytes	Latency	Connect Time(...)
1	18:55:47.612	Prueba de Car...	10 /ibk/sit/api/n...	1341	✓	840	1067	1341	28
2	18:55:48.958	Prueba de Car...	10 /ibk/sit/api/n...	536	✓	840	1070	536	41
3	18:55:49.497	Prueba de Car...	10 /ibk/sit/api/n...	1932	✓	840	1066	1932	1461
4	18:55:51.433	Prueba de Car...	10 /ibk/sit/api/n...	522	✓	840	1070	522	54
5	18:55:51.964	Prueba de Car...	10 /ibk/sit/api/n...	561	✓	840	1073	561	37
6	18:55:52.527	Prueba de Car...	10 /ibk/sit/api/n...	517	✓	840	1070	517	27
7	18:55:53.045	Prueba de Car...	10 /ibk/sit/api/n...	540	✓	840	1068	540	29
8	18:55:53.587	Prueba de Car...	10 /ibk/sit/api/n...	493	✓	840	1071	493	31
9	18:55:54.083	Prueba de Car...	10 /ibk/sit/api/n...	476	✓	840	1069	476	39
10	18:55:54.561	Prueba de Car...	10 /ibk/sit/api/n...	502	✓	840	1070	502	42

Figura 70. Lista de los tiempos de muestra de las ejecuciones realizadas (Elaborado por la investigadora).

Reporte resumen

Nombre: Reporte resumen

Comentarios

Escribir todos los datos a Archivo

Nombre de archivo Navegar... Log/Mostrar sólo: Escribir en Log Sólo Errores Éxitos

Etiqueta	# Muestras	Media	Min	Máx	Desv. Estándar	% Error	Rendimiento	Kb/sec	Sent KB/sec	Media de Bytes
10 /lbk/sit/api...	10	742	476	1932	466.94	0.00%	1.3/sec	1.10	1.40	840.0
Total	10	742	476	1932	466.94	0.00%	1.3/sec	1.10	1.40	840.0

Figura 71. Reporte de resumen de la prueba de carga ejecutada (Elaborado por la investigadora).

Tabla 20

Resumen de la prueba de carga al servicio API.

Etiqueta	Muestras	Media (ms)	Min (ms)	Máx (ms)	Desv. Estándar	% Error	Rendimiento	Media de Bytes
sms-not-transaccional	10	742	476	1932	466.93	0.0	1.34	1.10
Total	10	742	476	1932	466.93	0.0	1.34	1.10

Fuente: Autoría propia

2.3.5.1.5. Ejecución de Prueba de Concurrencia al Servicio REST.

Se ejecutó una prueba de resistencia al servicio API para envíos de mensajes SMS a través del proveedor Watson, para la prueba se utilizó la JMETER versión 5.0, a continuación, se detallan los resultados de la prueba.

Archivos de configuración.

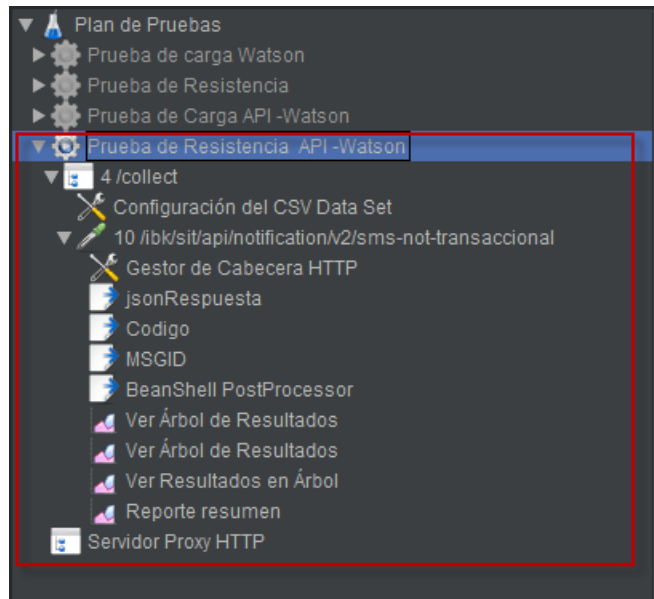


Figura 72. Archivo de configuración del script de Concurrency (Elaborado por la investigadora).

Se verifica que las ejecuciones de la prueba hayan sido exitosas.

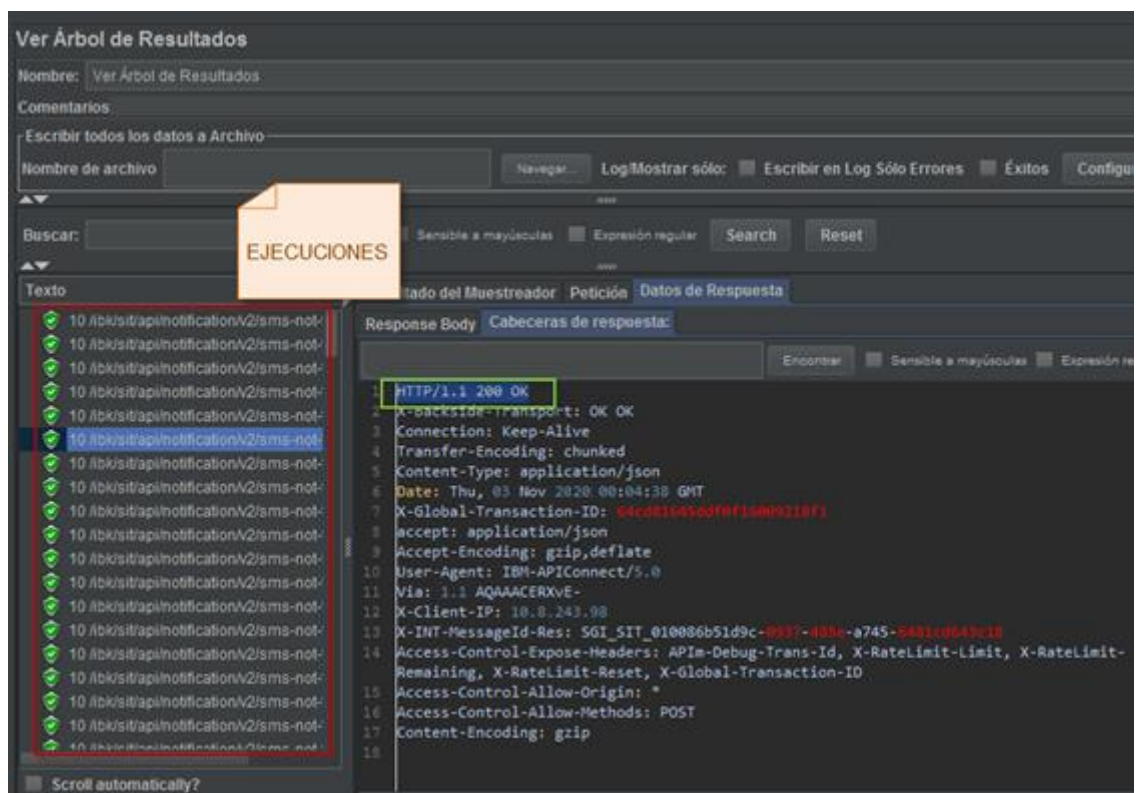


Figura 73. Lista de las ejecuciones realizadas (Elaborado por la investigadora).

Se realizó la ejecución de carga con 05 usuarios concurrentes durante 05 minutos.

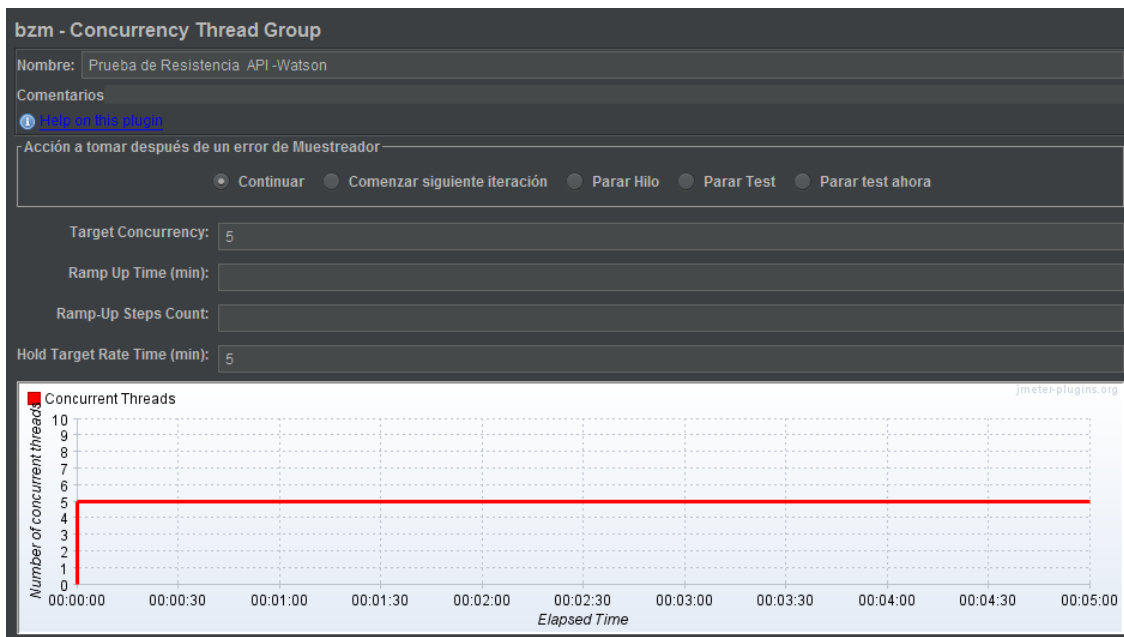


Figura 74. Configuración del grupo de concurrencia para la ejecución de carga (Elaborado por la investigadora).

Los tiempos de Muestra de cada una de las ejecuciones se detallan en la siguiente imagen:

Ver Resultados en Árbol

Nombre: Ver Resultados en Árbol

Comentarios

Escribir todos los datos a Archivo

Nombre de archivo Navegar...

Log/Mostrar sólo: Escribir en Log Sólo Errores Éxitos Configurar

Muestra #	Tiempo de co...	Nombre del hilo	Etiqueta	Tiempo de Mu...	Estado	Bytes	Sent Bytes	Latency	Connect Time(...)
1	19:04:37.684	Prueba de Re...	10 libk/sit/apil...	933	✓	703	1004	933	28
2	19:04:37.670	Prueba de Re...	10 libk/sit/apil...	983	✓	703	1001	983	28
3	19:04:37.674	Prueba de Re...	10 libk/sit/apil...	1012	✓	703	1004	1012	33
4	19:04:37.679	Prueba de Re...	10 libk/sit/apil...	1063	✓	703	1000	1063	25
5	19:04:37.690	Prueba de Re...	10 libk/sit/apil...	1268	✓	703	1007	1268	28
6	19:04:38.744	Prueba de Re...	10 libk/sit/apil...	632	✓	703	1003	632	34
7	19:04:38.689	Prueba de Re...	10 libk/sit/apil...	700	✓	703	1005	700	31
8	19:04:38.657	Prueba de Re...	10 libk/sit/apil...	835	✓	703	1002	835	32
9	19:04:38.960	Prueba de Re...	10 libk/sit/apil...	798	✓	703	1004	798	25
10	19:04:38.622	Prueba de Re...	10 libk/sit/apil...	1249	✓	703	1004	1249	48
11	19:04:39.391	Prueba de Re...	10 libk/sit/apil...	636	✓	703	1004	636	31
12	19:04:39.380	Prueba de Re...	10 libk/sit/apil...	664	✓	703	1001	664	32
13	19:04:39.495	Prueba de Re...	10 libk/sit/apil...	713	✓	703	1000	713	31
14	19:04:39.761	Prueba de Re...	10 libk/sit/apil...	545	✓	703	1004	545	43
15	19:04:39.874	Prueba de Re...	10 libk/sit/apil...	726	✓	703	1007	726	46
16	19:04:40.048	Prueba de Re...	10 libk/sit/apil...	704	✓	703	1002	704	33
17	19:04:40.029	Prueba de Re...	10 libk/sit/apil...	763	✓	703	1004	763	37
18	19:04:40.211	Prueba de Re...	10 libk/sit/apil...	607	✓	703	1005	607	41
19	19:04:40.308	Prueba de Re...	10 libk/sit/apil...	704	✓	703	1003	704	45
20	19:04:40.602	Prueba de Re...	10 libk/sit/apil...	594	✓	703	1004	594	37
21	19:04:40.753	Prueba de Re...	10 libk/sit/apil...	603	✓	703	1001	603	36
22	19:04:40.794	Prueba de Re...	10 libk/sit/apil...	659	✓	703	1004	659	26
23	19:04:40.819	Prueba de Re...	10 libk/sit/apil...	638	✓	703	1000	638	27
24	19:04:41.014	Prueba de Re...	10 libk/sit/apil...	693	✓	703	1004	693	35
25	19:04:41.459	Prueba de Re...	10 libk/sit/apil...	721	✓	703	1005	721	24
26	19:04:41.199	Prueba de Re...	10 libk/sit/apil...	1134	✓	703	1007	1134	44

Scroll automatically? Child samples? No. de Muestras 1975 Última Muestra 1206 Medida 757 Desviación 278

Figura 75. Lista de los tiempos de muestra de las ejecuciones realizadas. (Elaborado por la investigadora).

Como resultado de la prueba se obtuvo lo siguiente:

Con 05 usuarios concurrentes durante 05 minutos se realizaron 1,975 ejecuciones, con una media de ejecución de 757 milisegundos y con un tiempo mínimos de 429 ms y máximos de 3,878 milisegundos.

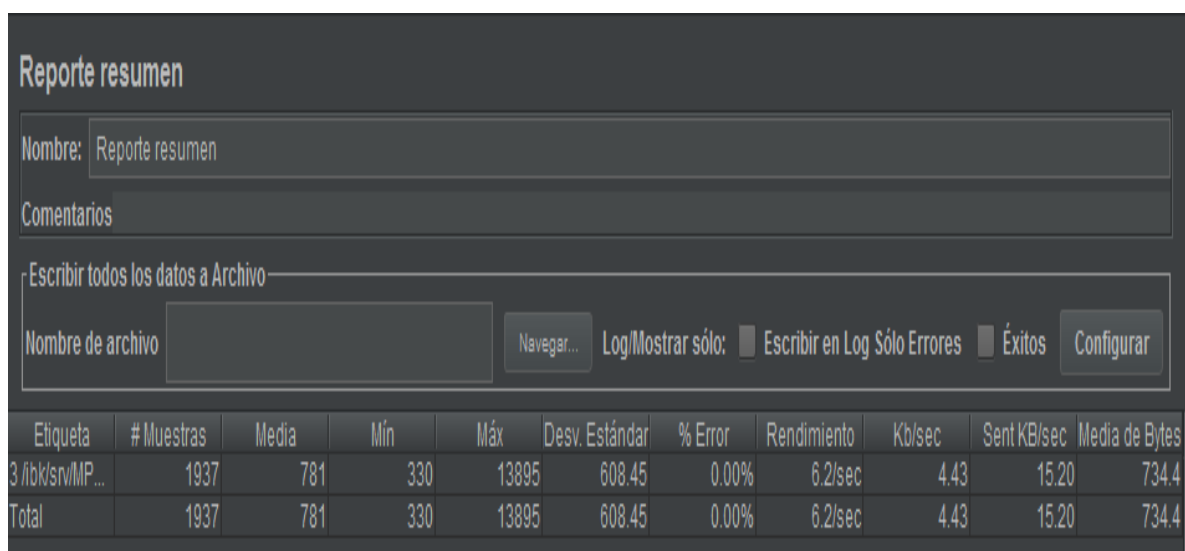


Figura 76. Reporte de resumen de la prueba de concurrencia ejecutada (Elaborado por la investigadora).

Tabla 21

Resumen de la prueba de concurrencia al servicio API.

Etiqueta	Muestras	Media (ms)	Min (ms)	Máx (ms)	Desv. Estándar	% Error	Rendimiento	Media de Bytes
sms-not-transaccional	1975	757	429	3878	278.71	0.0	6.57	4.51
Total	1975	757	429	3878	278.71	0.0	6.57	4.51

Fuente: Autoría propia

Tabla 22

Evaluación de los tiempos de respuesta según la plataforma

Plataforma	Muestras	Media (ms)	Min (ms)	Máx (ms)	Desv. Estandar	% Error	Rendimiento	Media de Bytes
IBM INTEGRATION BUS	1937	781	330	13895	608.45	0	6.18	734.44
IBM API CONNECT	1975	757	429	3878	278.71	0.0	6.57	4.51

Fuente: Autoría propia

Interpretación de Datos

Porcentaje de diferencia

$$\text{Respuesta Media} = \frac{(757-781)}{781} = -3.1\%$$

$$\text{Respuesta Máxima} = \frac{(3878-13895)}{13895} = -72.1\%$$

La media de respuesta del BUS es de 781 ms, mientras que el API es de 757 ms, se visualiza una ligera diferencia del 3.1% entre la respuesta de las plataformas.

Se visualiza que el BUS tuvo un tiempo máximo de respuesta de 13895 ms; sin embargo, el API tuvo una respuesta máxima de 3878 ms., donde se evidencia una diferencia del 72.1%

2.3.5.1.6. Tiempo de despliegue.

Con la ejecución del Servicio SOAP se obtuvo una media de ejecución de 781 milisegundos, lográndose ejecutar 1937 llamadas al servicio en un periodo de 10 minutos

Con la ejecución del Servicio REST se obtuvo una media de ejecución de 757 milisegundos, lográndose ejecutar 1975 llamadas al servicio en un periodo de 10 minutos.

Tabla 23

Evaluación del tiempo de despliegue.

Servicio	Media Ejecución (ms)	# Muestras
SOAP	781	1937
REST	757	1975
Diferencia	24	38
Porcentaje	3.07%	1.96%

Fuente: Autoría propia

Interpretación de Datos.

De la evaluación de la categoría de tiempo de despliegue se obtuvo que la media de ejecución del servicio SOAP fue de 781 ms y la media del servicio REST fue de 757 ms, en la que se evidencia una diferencia de 24 segundos correspondiente al 3.07%.

Con respecto a las muestras se obtuvo una diferencia del 1.96% del servicio REST con un mayor número de respuestas respecto al servicio SOAP.

2.3.5.1.7. Utilización de Recursos.

Para la métrica de utilización de recursos se realizó la siguiente evaluación:

Líneas de Código. Para el Servicio SOAP, se necesitaron 6922 líneas de código que se encuentran distribuidas en los archivos:

- ESQL
- Config de BUS
- Monitor
- Firma del Servicio WSDL

Para el servicio REST se necesitaron 1425 líneas de código que se encuentran escritas en el archivo YAML.

Tabla 24

Evaluación de la cantidad de líneas de Código.

Servicio	Cantidad de Líneas de Código
SOAP	6922
REST	1425
<i>Diferencia</i>	<i>5497</i>
<i>Porcentaje</i>	<i>79%</i>

Fuente: Autoría propia

Interpretación de Datos

Con respecto a la métrica de cantidad de líneas de código se obtuvo una diferencia del 79%, evidenciando que para el servicio REST se utilizaron menos líneas de código respecto al servicio SOAP.

2.3.5.1.8. Tamaño de Espacio de almacenamiento.

El tamaño del Servicio SOAP, es de 7.04 MB distribuidos en los archivos:

- BAR
- Config de BUS
- Monitor
- Firma del Servicio WSDL

El tamaño del Servicio SOAP es de 0.016 MB correspondiente al archivo YAML.

Tabla 25

Evaluación de la cantidad de espacio de almacenamiento.

Servicio	Almacenamiento (MB)
SOAP	7.04
REST	0.016
Diferencia	7.024
Porcentaje	99.8%

Fuente: Autoría propia

Interpretación de Datos

Con respecto a la métrica de cantidad de espacio de almacenamiento se visualiza una diferencia del 99.8%, en el que se evidencia que con los archivos utilizados por el servicio REST se ocupa menos almacenamiento que los que se ocupan con el servicio SOAP.

2.4. Resultados

2.4.1. Interpretación del análisis

2.4.1.1. Líneas de Código.

Para el Servicio SOAP, se necesitaron 6922 líneas de código que se encuentran distribuidas en los archivos:

- ESQL
- Config de BUS
- Monitor
- Firma del Servicio WSDL

Para el servicio REST se necesitaron 1425 líneas de código que se encuentran escritas en el archivo YAML.

Tabla 26

Cantidad de Líneas de Código.

Líneas de código	
Servicio	Cantidad de Líneas de Código
SOAP	6922
REST	1425
<i>Diferencia</i>	5497
<i>Porcentaje</i>	79%

Fuente: Autoría propia

Se comprueba que el Servicio REST ha mejorado en un 79% en la cantidad de líneas de código respecto al servicio SOAP.

2.4.1.2. Tiempos de Programación.

Para el desarrollo del Servicio SOAP, se emplearon 40 horas-hombre

Para el servicio REST se necesitaron 24 horas-hombre.

Tabla 27

Tiempos de Programación.

Tiempos de Programación	
Servicio	Tiempo (Horas-hombre)
SOAP	40
REST	24
<i>Diferencia</i>	<i>16</i>
<i>Porcentaje</i>	<i>40%</i>

Fuente: Autoría propia

Se comprueba que el Servicio REST ha mejorado en un 40% en los tiempos de programación con una reducción de 16 horas-hombre respecto al servicio SOAP.

2.4.1.3. **Tamaño de Espacio de almacenamiento.**

El tamaño del Servicio SOAP, es de 7.04 MB distribuidos en los archivos:

- BAR
- Config de BUS
- Monitor
- Firma del Servicio WSDL

El tamaño del Servicio SOAP es de 0.016 MB correspondiente al archivo YAML.

Tabla 28

Tamaño de Espacio de almacenamiento.

Tamaño de Espacio de almacenamiento	
Servicio	Almacenamiento (MB)
SOAP	7.04
REST	0.016
<i>Diferencia</i>	<i>7.024</i>
<i>Porcentaje</i>	<i>99.8%</i>

Fuente: Autoría propia

Se comprueba que el Servicio REST ha mejorado en un 99.8% en el espacio de almacenamiento respecto al servicio SOAP.

2.4.2. Discusión de los Resultados

De los resultados de la evaluación de la métrica de alcance en la Tabla 11, se demostró que con la plataforma IBM Integration BUS solo 05 canales pueden consumir los servicios de integración; sin embargo, con IBM API Connect se aumenta la posibilidad de que más canales consuman los servicios.

Con respecto a los resultados de las métricas de calidad, obtenidos de la puntuación realizada por los expertos que utilizan la plataforma de integración se demostró que en la categoría Confiabilidad existió un 33.3% de mejora de la plataforma API con respecto a BUS, para la categoría Funcionalidad se obtuvo 51.7% de diferencia entre la plataforma API y BUS y por último en la categoría Usabilidad se encontró un 100% de mejora de la plataforma IBM API Connect en comparación con la plataforma BUS, por lo que se demostró la mejora esta métrica con la implementación de IBM API Connect.

Para la métrica de tiempo se analizó el tiempo de respuesta del servicio y el tiempo de despliegue, de los resultados obtenidos se evidenció una diferencia del 3.1% en la media de ejecución, un 72.1% en los tiempos máximos de ejecución y una diferencia del 3.07% en el tiempo de despliegue lo que demostró que con la implementación de la plataforma

IBM API Connect existe una mejora en los tiempos respecto a la plataforma BUS.

De la evaluación de la métrica de utilización de recursos se obtuvo un obtuvo un 40% menos de líneas de código en el archivo REST en relación al archivo SOAP. Asimismo, se evidenció la reducción de 16 horas-hombre en el tiempo de programación del servicio REST respecto al servicio SOAP y finalmente se obtuvo un archivo REST con un 98% menos de espacio de almacenamiento en comparación con el archivo SOAP lo que evidencia una mejora de esta métrica con la implementación de IBM API Connect.

CONCLUSIONES

Para evaluar las plataformas de integración se realizó la medición de artefactos teniendo en cuenta diferentes criterios asociados los cuales fueron alcance, tiempo, calidad y utilización de recursos.

Los resultados obtenidos de las evaluaciones de las métricas aplicadas a los servicios REST respecto a los servicios SOAP evidenciaron lo siguiente:

- Se tiene un mayor alcance de canales que puedan ser consumidos por los servicios de integración.
- La apreciación de la calidad basado en la confiabilidad, funcionalidad y usabilidad es mejor en los servicios REST
- Los tiempos de despliegue de los servicios se reducen al utilizar servicios REST en los diferentes ambientes de la plataforma de Integración.
- La elaboración del servicio REST se desarrolla con menos líneas de código, que a su vez reducen la carga laboral.
- Para la programación de un servicio REST se consumen menos horas-hombre.
- Con el desarrollo de servicios REST se optimiza el almacenamiento de los archivos en cantidad de archivo y espacio.

Con el análisis de la evaluación de las métricas se comprobó que la implementación de la herramienta IBM API CONNECT V.5.0 en el banco Interbank permitió evidenciar una mejora en los desarrollos de la plataforma de integración de los web services.

RECOMENDACIONES

- Replicar la implementación de los servicios Rest reemplazando a los servicios Soap con los que cuenta el banco ya que según el análisis se ha evidenciado una mejora en tiempos, almacenamiento y performance.
- Se recomienda el uso de esta herramienta como un valor agregado en los proyectos actuales y futuros, lo cual ayudará al cliente a tener una adecuada gestión de sus servicios mediante la arquitectura de integración.
- Se recomienda aprovechar el uso de la tecnología de API para facilitar el diálogo de la capa de integración del banco y su Core.
- Se recomienda analizar otras herramientas que también realizan integración vía API como Microsoft Azzure, Amazon Web servicie,
- Se recomienda brindar capacitaciones a los diferentes equipos del banco para desarrollo y pruebas de servicios Rest.

BIBLIOGRAFÍA

Anónimo (2007). Metodología XP. Ingeniería de Software: obolog, Recuperado de: <http://ingsoftware072301.obolog.es/metodologia-xp-2012877>

Anónimo (2016) ISO 9126, Recuperado de: http://iso9126uts.blogspot.com/p/i_8.html

Abud (2012) Calidad en la Industria del Software. La Norma ISO-9126, Recuperado de: https://www.academia.edu/43099704/Iso_9126

Baquero (2015) ¿Qué son los web services y qué tecnología usar en su desarrollo?, Recuperado de <https://www.arsys.es/blog/programacion/disenio-web/web-services-desarrollo/>

Bbvaopen4u (2016) API´S, Cuatro ventajas de usar una API en tu empresa, Recuperado de <https://bbvaopen4u.com/es/actualidad/cuatro-ventajas-de-usar-una-API-en-tu-empresa>

Blancarte (2017) SOAP vs REST ¿cuál es mejor? Recuperado de: <https://www.oscarblancarteblog.com/2017/03/06/soap-vs-rest-2/>

Burgos, L. (2017). Análisis y evaluación de las arquitecturas REST Y SOAP, para el desarrollo de servicios web aplicados al ERP “ADRISERP” y su versión móvil en Android, Tesis para optar el título profesional de Ingeniero de Sistemas. Universidad Señor de Sipán – Chiclayo,

Cibergestión (2018), El valor de las APIs en el sector financiero, Recuperado de : <http://cibergestion.com/pe/valor-las-APIs-en-sector-financiero/>

Chávez, V. (2010). Sistema de Información para el control, Seguimiento y Mantenimiento del equipamiento hospitalario. Ingeniero Informático. Ricardo Palma.

Glickenhause (2019) ¿Qué es API Management?, Recuperado de:
<https://developer.ibm.com/APIconnect/2019/01/10/what-is-API-management/>

Gonzales, A (2019), IBM API CONNECT, ¿Qué es API Connect de IBM?, Recuperado de: https://www.cloudappi.net/en_US/blog/our-blog-1/post/ibm-API-connect-101

Hipertexto.Info (2005) XML Recuperado de:
<http://www.hipertexto.info/documentos/xml.htm>

IBM (s.f.) Conozca qué es una API y administre sus API´s con API Connect, Recuperado de <https://www.ibm.com/ar-es/cloud/API-connect>

IBM (s.f.) Getting Started, Recuperado de:
<https://developer.ibm.com/APIconnect/getting-started/>

IBM (s.f.) Roles de usuario de API Connect, Recuperado de:
https://www.ibm.com/support/knowledgecenter/SSMNE5.0.0/com.ibm.APIC.Overview.doc/overview_APIimgmt_users.html#overview_APIimgmt_users__APIIm_manager_roles

IBM (s.f.) Gestión de API en el sector bancario, Recuperado de:
<https://www.ibm.com/ar-es/cloud/API-connect/manage>

IBM (2017) Estrategia y terminología de empaquetado en API Connect, Recuperado de:

https://www.ibm.com/support/knowledgecenter/es/SSMNED_5.0.0/com.ibm.APIc.overview.doc/cAPIm_overview_orgsprodsplansAPIs.html#cAPIm_overview_orgsprodsplansAPIs__plans_and_products

IBM (2017) Visión general de IBM API Connect, Recuperado de:

https://www.ibm.com/support/knowledgecenter/es/SSMNED_5.0.0/com.ibm.APIc.overview.doc/APIc_capabilities.png

IBM (2020) Administrar sus API'S, Recuperado de :

https://www.ibm.com/support/knowledgecenter/en/SSMNED_2018/com.ibm.APIc.APIonprem.doc/APIonPrem_gettingstarted.html

IBM (2020) API Gateway, Recuperado de: <https://cloud.ibm.com/docs/API-gateway?topic=API-gateway-getting-started&locale=es>

IBM Knowledge Center (2014) ¿Qué es un Soap? Recuperado de:

https://www.ibm.com/support/knowledgecenter/es/SSKM8N_8.0.0/com.ibm.etools.mft.doc/ac55770_.htm

IBM Knowledge Center (2014) ¿Qué es WSDL? Recuperado de:

https://www.ibm.com/support/knowledgecenter/es/SSMKHH_9.0.0/com.ibm.etools.mft.doc/ac34640_.htm

Ionos(2020) Web services: servicios de máquina a máquina, Recuperado de:

<https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/web-services/>

Kocen(2017) Introducción a API Designer en API Connect, Recuperado de :

<https://developer.ibm.com/APIconnect/2017/05/10/API-designer-API-connect/>

Lázaro (2018) , Introducción a los Web Services, Recuperado de :

<https://diego.com.es/introduccion-a-los-web-services>

León (2008). ERP Demystified [Segunda Edición]. Ed. Tata McGraw-Hill. Nueva Deli.

Library.Gxtechnical (2008) WEB SERVICES Recuperado de:

<http://library.gxtechnical.com>

Medina, L., Pinedo, L. (2010). Implementación de un sistema de integración para las bibliotecas municipales de Lima y Callao utilizando SOA y J2ME
Tesina para optar el título profesional de Ingeniero de sistemas.
Ingeniero De Sistemas. Universidad Nacional Mayor de San Marcos.

Murguía (2016) Planeamiento estratégico del banco internacional del Perú

S.A.A.– Interbank 2015-2016, Recuperado de:

https://www.academia.edu/36779370/Tesis_para_optar_el_T%C3%ADtulo_Profesional_de_Ingeniero_Empresarial_y_de_Sistemas

Oficina de consultoría en el sector de la ingeniería civil en lima Perú (2000)

Recuperado de: <https://www.gcaq.com.pe/proyectos/2000-torre-interbank.html>

Redhat (2018) ¿Qué es un API? Recuperado de:

<https://www.redhat.com/es/topics/API/what-are-application-programming-interfaces>

Redhat (2020) ¿Qué es un API? Qué son las API y para qué sirven,

Recuperado de: <https://www.redhat.com/es/topics/API/what-are-application-programming-interfaces>

Sagástegui, W (2010) ¿Qué es y para qué sirve el lenguaje de etiquetas XML

(Extensible Markup Language)?, Recuperado de:

https://www.aprenderaprogramar.com/index.php?option=com_content&view=article&id=102:ique-es-y-para-que-sirve-el-lenguaje-de-etiquetas-xml-extensible-markup-language&catid=46&Itemid=163

Sandy (2014), Conceptos básicos de servicios WEB, Recuperado de:

<http://desarrolloconsoa.blogspot.com/2014/02/conceptos-basicos-de-servicios-web-soap.html>

Sastre (2010) Norma ISO 9126:2001, Recuperado de

https://es.scribd.com/embeds/94681498/content?start_page=1&view_mode=scroll&access_key=key-DXFMtVntRav5tItoCCWR

Silva (s.f.), TMMi: Un modelo especializado en pruebas, Recuperado de:

<https://sg.com.mx/revista/31/tmmi-un-modelo-especializado-pruebas>.

Solcataleya (2016), Roles de la metodología XP, Recuperado de

<https://view.genial.ly/57ed1cd094fd6edca3556d/interactive-content-modelo-de-desarrollo-de-software>

Speedcheck (2016) HTTP Recuperado de:

<https://www.speedcheck.org/es/wiki/http/>

Universidad de Alicante (2014), Introducción a los Servicios Web. Invocación de servicios web SOAP, Recuperado de:

http://www.jtech.ua.es/j2ee/publico/servc-web-2012-13/sesion01-apuntes.html?fbclid=IwAR1RRp5IYLTCQvLZxV-_S_BS7cELnFjcZ5xndCZNbXSeFwTeF9uAyA7HSpM#Arquitectura+de+los+Servicios+Web

Valencia, D. (2018). Análisis de frameworks de desarrollo De API Rest y su impacto en el rendimiento de aplicaciones web con arquitectura Spa. Ingeniero. Universidad Técnica del Norte.

ANEXOS

ANEXO 1

HISTORIA DE CAMBIOS

CONTROL DE CONFIGURACIÓN

Título:	Estructura de ensamble del API
Autor:	Karla Vicente
Fecha:	25/09/2019

Histórico de Versiones

Versión	Fecha	Estado	Responsable	Descripción del Cambio
1.0	25/09/2019		Karla Vicente	Creación del Documento

Figura 77 Estructura de Historia de Cambios Base (Elaborado por la investigadora)

ANEXO 3

	ACTA DE REUNIÓN	Proyecto: Actualizar Plataforma de Integración

Fecha:	17 de Noviembre 2019	Ubicación:	Sala 4A2 - Piso 4 – Torre A
Hora Inicio:	8:00 am	Hora Fin:	11:00 am

PARTICIPANTES:

Nombre	Área /Proveedor	¿Asistió?	Iniciales
Soledad Perrone	Gigatek	Sí	SP
Rubén Mejía	Synopsis	Sí	RM
Giancarlo Chávez	Synopsis	No	GC
Karla Vicente	Synopsis	Sí	KV
Juan Manuel Cuya	DCO_IBM	Sí	JMC
Martín Navarro	DCO_IBM	No	MN
Saulo Pires	DCO_IBM	No	SPI

Sylvia Nakaya	Ingeniería	Sí	SN
Eduardo Zúñiga	Infraestructura	No	EZ
Patricia Gutierrez	Arquitectura	No	PG
José Castillo	Líder técnico	Sí	JC
Sandro Reátegui	Especialista de sistemas	Si	SR
Ofelia Pardo-Figueroa	Gestor de proyecto	Si	OPF

Objetivo de la reunión:

Elaborar el plan de trabajo detallado de las actividades pendientes para Actualizar la plataforma de Integración

Temas tratados:

1. KV presentó detalle de la solución propuesta para que el equipo de proyecto maneje el mismo nivel de información.
2. JC comentó temas pendientes como la solución de back up y la réplica de datos en z.
3. SR comentó la necesidad de contar con una solución de Monitoreo.
4. JMC planteó la idea de contar con una solución temporal de back up a nivel de objetos.
5. SR confirmó que sí es necesaria la instalación de IBM Omegamon para el monitoreo.

Acuerdos:

- a) Sí se requiere 10 ambientes según diseño mostrado en la presentación
- b) Hoy tenemos en el CCA 4 ambientes (UAT (2)+SIT+DES) ya instalados por Gigatek
- c) Está pendiente de instalar 4 ambientes para PRD en CCP.

- d) Se necesita la cotización de Contingencia.
- e) Se necesita definir la solución de monitoreo.
- f) Se requiere un back up temporal.

Compromisos:

Acción	Responsable(s)	Estado	Fecha de Finalización
Consultar por la solución de back up	Juan Manuel Cuya	En proceso	10.12
Consultar por la solución de réplica de datos en z	Juan Manuel Cuya	En proceso	10.12
Cotización del servicio de contingencia	Soledad Perrone	Pendiente	12.12
Consultar por solución de monitoreo por DCOBPM o Spectrum	Juan Manuel Cuya	En proceso	12.12
Confirmar que IBK cuenta con la licencia de Omengamon	Sandro Reátegui	En proceso	14.12
Obtener el backup de productos a nivel de objetos de lo que tenemos instalado a hoy	Karla Vicente/KV	Pendiente	14.12
Finalizar los 4 ambientes en CCA	Rubén Mejía/JC	En proceso	14.12
Solicitar a Daniel Villafuerte las 2 IP pendientes para los 2 ambientes UAT del CCA	José Castillo	Pendiente	22.12
Finalizar desarrollo de la plataforma de Integración de propuesta	Karla Vicente/KV	En proceso	15.12
Finalizar los 2 ambientes UAT del CCA	Soledad Perrone	Pendiente	14.12
Definir proceso de certificación	José Castillo	Pendiente	14.12

Envío de requerimientos técnicos a Sylvia	Soledad Perrone	Pendiente	14.12
Plan de trabajo para 4 ambientes PRD del CCP	Soledad Perrone	Pendiente	14.12

De ser enviado el acta por correo, si no contamos con ninguna observación en 14 horas por todos los participantes se da por conforme a todos los acuerdos establecidos en el acta.

Los estados de los compromisos son:

1. **Pendiente:** Si aún no se inicia a gestionar o desarrollo el compromiso.
2. **Finalizado:** Si se cumplió con el compromiso.
3. **En proceso:** Si el compromiso está en curso.