



**Universidad Nacional Mayor de San Marcos**

**Universidad del Perú. Decana de América**

Dirección General de Estudios de Posgrado

Facultad de Ingeniería de Sistemas e Informática

Unidad de Posgrado

**Framework para la externalización de la configuración  
en entornos multitenant**

**TESIS**

Para optar el Grado Académico de Magíster en Ingeniería de  
Sistemas e Informática con mención en Ingeniería de Software

**AUTOR**

Jose Miguel MORI PEÑA

**ASESOR**

Mg. Fany Yexenia SOBERO RODRÍGUEZ

Lima, Perú

2025



Reconocimiento - No Comercial - Compartir Igual - Sin restricciones adicionales

<https://creativecommons.org/licenses/by-nc-sa/4.0/>

Usted puede distribuir, remezclar, retocar, y crear a partir del documento original de modo no comercial, siempre y cuando se dé crédito al autor del documento y se licencien las nuevas creaciones bajo las mismas condiciones. No se permite aplicar términos legales o medidas tecnológicas que restrinjan legalmente a otros a hacer cualquier cosa que permita esta licencia.

## Referencia bibliográfica

---

Mori, J. (2025). *Framework para la externalización de la configuración en entornos multitenant*. [Tesis de maestría, Universidad Nacional Mayor de San Marcos, Facultad de Ingeniería de Sistemas e Informática, Unidad de Posgrado]. Repositorio institucional Cybertesis UNMSM.

---

### Metadatos complementarios

| Datos de autor                   |                                                                                           |
|----------------------------------|-------------------------------------------------------------------------------------------|
| Nombres y apellidos              | Jose Miguel Mori Peña                                                                     |
| Tipo de documento de identidad   | DNI                                                                                       |
| Número de documento de identidad | 43359119                                                                                  |
| URL de ORCID                     | <a href="https://orcid.org/0000-0002-0667-3869">https://orcid.org/0000-0002-0667-3869</a> |
| Datos de asesor                  |                                                                                           |
| Nombres y apellidos              | Fany Yexenia Sobero Rodríguez                                                             |
| Tipo de documento de identidad   | DNI                                                                                       |
| Número de documento de identidad | 20120467                                                                                  |
| URL de ORCID                     | <a href="https://orcid.org/0000-0002-0323-6110">https://orcid.org/0000-0002-0323-6110</a> |
| Datos del jurado                 |                                                                                           |
| Presidente del jurado            |                                                                                           |
| Nombres y apellidos              | Ciro Rodriguez Rodriguez                                                                  |
| Tipo de documento                | DNI                                                                                       |
| Número de documento de identidad | 06020241                                                                                  |
| Miembro del jurado 1             |                                                                                           |
| Nombres y apellidos              | Percy Edwin De la Cruz Vélez de Villa                                                     |
| Tipo de documento                | DNI                                                                                       |
| Número de documento de identidad | 08583141                                                                                  |
| Miembro del jurado 2             |                                                                                           |
| Nombres y apellidos              | Maria Elena Ruiz Rivera                                                                   |
| Tipo de documento                | DNI                                                                                       |
| Número de documento de identidad | 08960021                                                                                  |
| Datos de investigación           |                                                                                           |

|                                                        |                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|--------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Línea de investigación                                 | C.0.3.22 Ingeniería de Software                                                                                                                                                                                                                                                                                                                                                                                                            |
| Grupo de investigación                                 | No aplica.                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Agencia de financiamiento                              | Sin financiamiento.                                                                                                                                                                                                                                                                                                                                                                                                                        |
| Ubicación geográfica de la investigación               | País: Perú<br>Departamento: Lima<br>Provincia: Lima<br>Distrito: Lima<br>Latitud: -12.056445<br>Longitud: -77.085994                                                                                                                                                                                                                                                                                                                       |
| Año o rango de años en que se realizó la investigación | 2023 - 2024                                                                                                                                                                                                                                                                                                                                                                                                                                |
| URL de disciplinas OCDE                                | <p>Ciencias de la computación<br/><a href="https://purl.org/pe-repo/ocde/ford#1.02.01">https://purl.org/pe-repo/ocde/ford#1.02.01</a></p> <p>Ciencias de la información<br/><a href="https://purl.org/pe-repo/ocde/ford#1.02.02">https://purl.org/pe-repo/ocde/ford#1.02.02</a></p> <p>Ingeniería de sistemas y comunicaciones<br/><a href="https://purl.org/pe-repo/ocde/ford#2.02.04">https://purl.org/pe-repo/ocde/ford#2.02.04</a></p> |



UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS  
Universidad del Perú. Decana de América  
Facultad de Ingeniería de Sistemas e Informática  
Vicedecanato de Investigación y Posgrado  
Unidad de Posgrado

**SUSTENTACIÓN DE TESIS PARA OPTAR EL GRADO ACADÉMICO DE MAGÍSTER  
EN INGENIERÍA DE SISTEMAS E INFORMÁTICA CON MENCIÓN EN INGENIERÍA  
DE SOFTWARE**

A los veinte (20) días del mes de agosto de 2025, siendo las 14:00 pm., se reunieron en el Auditorio Alfredo Celso Alva Bravo, el Jurado de Tesis conformado por los siguientes docentes:

Dr. Ciro Rodriguez Rodriguez (Presidente)  
Mg. Percy Edwin De la Cruz Vélez de Villa (Miembro)  
Mg. Maria Elena Ruiz Rivera (Miembro)  
Mg. Fany Yexenia Sobero Rodríguez (Miembro Asesor)

Se inició la Sustentación invitando al candidato a Magíster **JOSE MIGUEL MORI PEÑA**, para que realice la exposición oral de la tesis para optar el Grado Académico de Magíster en Ingeniería de Sistemas e Informática con mención en Ingeniería de Software, siendo la Tesis intitulada:

**"Framework para la externalización de la configuración en entornos multitenant".**

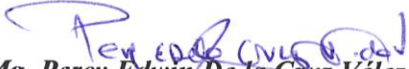
Concluida la exposición, los miembros del Jurado de Tesis procedieron a formular sus preguntas que fueron absueltas por el graduando; acto seguido se procedió a la evaluación correspondiente, habiendo obtenido la siguiente calificación:

..... *Quince* ..... (15) .....

Por tanto, el presidente del Jurado, de acuerdo con el Reglamento General de Estudios de Posgrado, otorga al Bachiller **JOSE MIGUEL MORI PEÑA** el Grado Académico de Magíster en Ingeniería de Sistemas e Informática con mención en Ingeniería de Software.

Siendo las *15:50* horas, el presidente del Jurado de Tesis, da por concluido el acto académico de Sustentación de Tesis.

  
Dr. Ciro Rodriguez Rodriguez  
(Presidente)

  
Mg. Percy Edwin De la Cruz Vélez de Villa  
(Miembro)

  
Mg. Maria Elena Ruiz Rivera  
(Miembro)

  
Mg. Fany Yexenia Sobero Rodríguez  
(Miembro Asesor)



UNIVERSIDAD NACIONAL MAYOR DE

**SAN MARCOS**

Universidad del Perú, Decana de América

### Anexo 1

## CERTIFICADO DE SIMILITUD

Yo, Fany Yexenia Sobero Rodríguez en mi condición de asesor acreditado con el Dictamen de la UPG N° 000254-2023-UPG-VDIP-FISI de la tesis, cuyo título es “Framework para la externalización de la configuración en entornos multitenant”, presentado por el bachiller Jose Miguel Mori Peña para optar el grado de magister en Ingeniería de Sistemas e Informática con mención en Ingeniería de Software, **CERTIFICO** que se ha cumplido con lo establecido en la Directiva de Originalidad y de Similitud de Trabajos Académicos, de Investigación y Producción Intelectual. Según la revisión, análisis y evaluación mediante la plataforma PIDS, el documento evaluado cuenta con el 15% de similitud, nivel **PERMITIDO** para continuar con los trámites correspondientes y para su **publicación en el repositorio institucional**.

Se emite el presente certificado en cumplimiento de lo establecido en las normas vigentes, como uno de los requisitos para la obtención del grado de magister en Ingeniería de Sistemas e Informática con mención en Ingeniería de Software.

Lima, 07 de noviembre de 2024

Firma del Asesor

Nombres y Apellidos del asesor: Fany Yexenia Sobero Rodríguez

DNI: 20120467

\* Según corresponda: bachiller, magíster, doctor; título profesional o título de segunda especialidad

*La vigencia del presente certificado corresponde a un período de noventa (90) días desde la fecha de su emisión; siempre que no haya observaciones en el texto durante el proceso de revisión por el jurado informante o surja alguna contingencia.*

### **Dedicatoria**

Con todo mi cariño y mi amor para las personas que hicieron todo en la vida para que pueda lograr mis sueños, por motivarme y darme la mano cuando sentía que el camino se terminaba, a ustedes por siempre mi corazón y mi agradecimiento.

### **Agradecimientos**

A mi asesora de tesis la Profesora Fany Sobero Rodríguez, por la orientación que me brindó para la realización y culminación de esta tesis.

A la profesora Nora La Serna, por su apoyo y amistad que me permitió aprender mucho más de lo estudiado en el proyecto.

A todos nuestros profesores de posgrado que nos enseñaron tanto de la profesión como de la vida, impulsándonos siempre a seguir adelante.

## Índice general

|                                                                     |    |
|---------------------------------------------------------------------|----|
| Lista de tablas .....                                               | 7  |
| Lista de figuras.....                                               | 9  |
| Resumen.....                                                        | 12 |
| Abstract.....                                                       | 13 |
| CAPITULO 1: INTRODUCCION .....                                      | 14 |
| 1.1.    Situación Problemática .....                                | 14 |
| 1.2.    Formulación del Problema .....                              | 15 |
| 1.2.1.    Problema General.....                                     | 15 |
| 1.2.2.    Problemas Específicos .....                               | 15 |
| 1.3.    Justificación Teórica .....                                 | 16 |
| 1.4.    Justificación Practica .....                                | 17 |
| 1.5.    Objetivos .....                                             | 18 |
| 1.5.1.    Objetivo General.....                                     | 18 |
| 1.5.2.    Objetivos Específicos.....                                | 18 |
| CAPITULO 2 MARCO TEORICO.....                                       | 19 |
| 2.1.    Marco Filosófico o Epistemológico de la Investigación ..... | 19 |
| 2.2.    Antecedentes de Investigación .....                         | 19 |
| 2.3.    Bases teóricas .....                                        | 39 |
| 2.3.1.    Técnicas de Externalización .....                         | 39 |
| 2.3.2.    Sistemas de software existentes. ....                     | 43 |
| 2.3.2.1.    Consul.....                                             | 43 |
| 2.3.2.2.    Etcad .....                                             | 44 |
| 2.3.2.3.    Archaius .....                                          | 44 |
| 2.3.2.4.    Zookeeper.....                                          | 45 |
| 2.3.2.5.    Spring Cloud Config .....                               | 46 |
| 2.3.2.6.    AWS AppConfig .....                                     | 46 |

|                                                  |                                                         |     |
|--------------------------------------------------|---------------------------------------------------------|-----|
| 2.3.2.7.                                         | Azure App Configuration .....                           | 47  |
| 2.3.3.                                           | Patrón Externalización de la Configuración .....        | 48  |
| 2.3.4.                                           | El Teorema de CAP .....                                 | 49  |
| 2.3.5.                                           | Arquitectura Multitenant .....                          | 50  |
| CAPITULO 3 METODOLOGIA.....                      |                                                         | 51  |
| 3.1.                                             | Tipo y diseño de la investigación .....                 | 51  |
| 3.2.                                             | Unidad de análisis.....                                 | 51  |
| 3.3.                                             | Población de estudio.....                               | 51  |
| 3.4.                                             | Tamaño de la muestra .....                              | 52  |
| 3.5.                                             | Selección de muestra .....                              | 52  |
| 3.6.                                             | Técnicas de recolección de datos .....                  | 52  |
| 3.7.                                             | Análisis e interpretación de la información.....        | 53  |
| 3.8.                                             | Metodología de desarrollo de software .....             | 54  |
| 3.9.                                             | Propuesta Solución .....                                | 56  |
| 3.10.                                            | Comparación de herramientas existentes.....             | 57  |
| CAPITULO 4 IMPLEMENTACION DE LA HERRAMIENTA..... |                                                         | 60  |
| 4.1.                                             | Implementación.....                                     | 60  |
| 4.1.1.                                           | Fase de Exploración .....                               | 60  |
| 4.1.2.                                           | Fase de Planificación.....                              | 75  |
| 4.1.4.                                           | Fase de Puesta a Producción.....                        | 102 |
| 4.1.5.                                           | Validación del software .....                           | 105 |
| CAPITULO 5 RESULTADOS Y DISCUSION.....           |                                                         | 110 |
| 5.1.                                             | Análisis, interpretación y discusión de resultados..... | 110 |
| 5.1.1.                                           | Objetivos de la validación de la herramienta .....      | 110 |
| 5.1.2.                                           | Diseño de la validación .....                           | 110 |
| 5.1.3.                                           | Ejecución de la validación.....                         | 111 |
| 5.2.                                             | Pruebas de hipótesis .....                              | 118 |

|                                                               |                                                     |     |
|---------------------------------------------------------------|-----------------------------------------------------|-----|
| 5.2.1.                                                        | Prueba de hipótesis del objetivo específico 1 ..... | 118 |
| 5.2.2.                                                        | Prueba de hipótesis del objetivo específico 2 ..... | 119 |
| 5.2.3.                                                        | Prueba de hipótesis del objetivo específico 3 ..... | 120 |
| 5.2.4.                                                        | Prueba de hipótesis del objetivo específico 4 ..... | 121 |
| 5.3.                                                          | Presentación de resultados.....                     | 122 |
| CONCLUSIONES .....                                            |                                                     | 123 |
| RECOMENDACIONES .....                                         |                                                     | 124 |
| REFERENCIAS BIBLIOGRAFICAS .....                              |                                                     | 125 |
| ANEXOS .....                                                  |                                                     | 129 |
| Anexo 1 Operacionalización de las variables.....              |                                                     | 129 |
| Anexo 2 Matriz de Consistencia.....                           |                                                     | 130 |
| Anexo 3 Encuesta de Satisfacción de usuario Post-Uso .....    |                                                     | 131 |
| Anexo 4 Historias de Usuario.....                             |                                                     | 132 |
| Anexo 5 Tareas de Programación .....                          |                                                     | 137 |
| Anexo 6 Capturas del taller de validación del framework ..... |                                                     | 145 |
| Anexo 7 Proyecto para el taller practico .....                |                                                     | 150 |
| Anexo 8 Materiales de capacitación .....                      |                                                     | 153 |

## Lista de tablas

|          |                                                             |     |
|----------|-------------------------------------------------------------|-----|
| Tabla 1  | Propiedades deseadas de la Arquitectura.....                | 21  |
| Tabla 2  | Frameworks de configuración estudiados y sus firmas.....    | 29  |
| Tabla 3  | Taxonomía de los 11 frameworks de configuración.....        | 30  |
| Tabla 4  | Proyectos usando solo el framework.....                     | 30  |
| Tabla 5  | Características compatibles de las metodologías ágiles..... | 54  |
| Tabla 6  | Análisis comparativo de herramientas.....                   | 59  |
| Tabla 7  | Lista historias de usuario.....                             | 60  |
| Tabla 8  | Historia de usuario 1.....                                  | 61  |
| Tabla 9  | Historia de usuario 2.....                                  | 62  |
| Tabla 10 | Historia de usuario 3.....                                  | 62  |
| Tabla 11 | Historia de usuario 4.....                                  | 63  |
| Tabla 12 | Tarea de programación 1.....                                | 75  |
| Tabla 13 | Tarea de programación 2.....                                | 76  |
| Tabla 14 | Tarea de programación 3.....                                | 76  |
| Tabla 15 | Tarea de programación 4.....                                | 77  |
| Tabla 16 | Lista priorizada de historias de usuario.....               | 78  |
| Tabla 17 | Lista estimada de historias de usuario.....                 | 79  |
| Tabla 18 | Descripción de la tabla "cloud_config_application".....     | 102 |
| Tabla 19 | Descripción de la tabla "cloud_config_profile".....         | 102 |
| Tabla 20 | Descripción de la tabla "cloud_config_tenant".....          | 103 |
| Tabla 21 | Descripción de la tabla "cloud_config_setting".....         | 103 |
| Tabla 22 | Descripción de la tabla "cloud_config_schema".....          | 103 |
| Tabla 23 | Resultado de la encuesta de satisfacción de usuario.....    | 112 |
| Tabla 24 | Alfa de Cronbach.....                                       | 112 |
| Tabla 25 | Resultados de la encuesta por escala.....                   | 113 |
| Tabla 26 | Comparación con valores normativos.....                     | 114 |
| Tabla 27 | Historia de usuario 5.....                                  | 132 |
| Tabla 28 | Historia de usuario 6.....                                  | 133 |
| Tabla 29 | Historia de usuario 7.....                                  | 133 |
| Tabla 30 | Historia de usuario 8.....                                  | 134 |
| Tabla 31 | Historia de usuario 9.....                                  | 135 |
| Tabla 32 | Historia de usuario 10.....                                 | 135 |
| Tabla 33 | Historia de usuario 11.....                                 | 136 |
| Tabla 34 | Tarea de programación 5.....                                | 137 |
| Tabla 35 | Tarea de programación 6.....                                | 137 |

|          |                                |     |
|----------|--------------------------------|-----|
| Tabla 36 | Tarea de programación 7 .....  | 138 |
| Tabla 37 | Tarea de programación 8 .....  | 138 |
| Tabla 38 | Tarea de programación 9 .....  | 139 |
| Tabla 39 | Tarea de programación 10 ..... | 139 |
| Tabla 40 | Tarea de programación 11 ..... | 140 |
| Tabla 41 | Tarea de programación 12 ..... | 140 |
| Tabla 42 | Tarea de programación 13 ..... | 141 |
| Tabla 43 | Tarea de programación 14 ..... | 141 |
| Tabla 44 | Tarea de programación 15 ..... | 142 |
| Tabla 45 | Tarea de programación 16 ..... | 142 |
| Tabla 46 | Tarea de programación 17 ..... | 143 |
| Tabla 47 | Tarea de programación 18 ..... | 143 |
| Tabla 48 | Tarea de programación 19 ..... | 144 |
| Tabla 49 | Tarea de programación 20 ..... | 144 |

## Lista de figuras

|           |                                                                          |    |
|-----------|--------------------------------------------------------------------------|----|
| Figura 1  | Algoritmo para la creación de la base de datos multitenant.....          | 20 |
| Figura 2  | Vista funcional del estilo propuesto .....                               | 22 |
| Figura 3  | Estrategias de almacenamiento de datos.....                              | 23 |
| Figura 4  | Esquemas separados y compartidos.....                                    | 24 |
| Figura 5  | Estrategia de almacenamiento preferida .....                             | 25 |
| Figura 6  | Criterio para determinar la estrategia de almacenamiento preferida ..... | 26 |
| Figura 7  | Arquitectura multitenant en la nube.....                                 | 28 |
| Figura 8  | Arquitectura del sistema propuesto .....                                 | 32 |
| Figura 9  | Gráfico comparativo de los tiempos de respuesta .....                    | 33 |
| Figura 10 | Gráfico comparativo de los tamaños de respuesta .....                    | 34 |
| Figura 11 | Arquitectura de la herramienta de interoperabilidad .....                | 35 |
| Figura 12 | Tiempo de retraso entre componentes en milisegundos .....                | 36 |
| Figura 13 | Tiempo de carga para 100000 registros en 5 base de datos.....            | 37 |
| Figura 14 | Tiempo de ejecución para 100000 registros en 5 bases de datos .....      | 38 |
| Figura 15 | Asignación de variables de entorno en un script de Linux .....           | 39 |
| Figura 16 | Ejemplo de configuración de archivos usando YAML.....                    | 41 |
| Figura 17 | Arquitectura de Consul.....                                              | 43 |
| Figura 18 | Características de Etcd .....                                            | 44 |
| Figura 19 | Zookeeper service.....                                                   | 45 |
| Figura 20 | Externalización de la configuración .....                                | 48 |
| Figura 21 | Las posibles opciones según el teorema de CAP.....                       | 49 |
| Figura 22 | Metodología XP.....                                                      | 55 |
| Figura 23 | Propuesta solución.....                                                  | 56 |
| Figura 24 | Arquitectura propuesta - vista lógica .....                              | 64 |
| Figura 25 | Arquitectura propuesta - vista de proceso (client).....                  | 65 |
| Figura 26 | Arquitectura propuesta - vista de proceso (server) .....                 | 66 |
| Figura 27 | Arquitectura propuesta - vista física .....                              | 67 |
| Figura 28 | Arquitectura propuesta - vista de desarrollo (server) .....              | 68 |
| Figura 29 | Arquitectura propuesta - vista de desarrollo (client).....               | 68 |
| Figura 30 | Arquitectura propuesta - escenarios.....                                 | 69 |
| Figura 31 | Estrategia de ramificación .....                                         | 70 |
| Figura 32 | Bitbucket - Lista de ramas.....                                          | 71 |
| Figura 33 | Bitbucket - Creación del Pull Request .....                              | 72 |
| Figura 34 | Bitbucket - Revisión de Código .....                                     | 72 |
| Figura 35 | Bitbucket - Integración continua .....                                   | 73 |

|           |                                                                        |     |
|-----------|------------------------------------------------------------------------|-----|
| Figura 36 | Sonarqube - Evaluación de código.....                                  | 74  |
| Figura 37 | Jira - Timeline .....                                                  | 74  |
| Figura 38 | Iteración 1 - Diagrama de clases.....                                  | 80  |
| Figura 39 | Iteración 2- Diagrama de clases.....                                   | 81  |
| Figura 40 | Iteración 3 - Diagrama de clases.....                                  | 82  |
| Figura 41 | Iteración 4 - Diagrama de clases.....                                  | 83  |
| Figura 42 | Iteración 5 - Diagrama de clases.....                                  | 84  |
| Figura 43 | Iteración 6- Diagrama de clases del "Configuration Server" .....       | 85  |
| Figura 44 | Iteración 6 - Diagrama de clases de "Configuration Client" .....       | 86  |
| Figura 45 | Iteración 7 - Diagrama de clases.....                                  | 86  |
| Figura 46 | Iteración 8 - Diagrama de clases del "Configuration Server" I .....    | 87  |
| Figura 47 | Iteración 8 - Diagrama de clases del "Configuration Server" II .....   | 88  |
| Figura 48 | Iteración 8 - Diagrama de clases del "Configuration Server" III .....  | 89  |
| Figura 49 | Iteración 8 - Diagrama de clases del "Configuration Client".....       | 89  |
| Figura 50 | Iteración 9 - Diagrama de clases del "Configuration Client".....       | 90  |
| Figura 51 | Iteración 9 - Diagrama de clases del "Configuration Server" I .....    | 91  |
| Figura 52 | Iteración 9 - Diagrama de clases del "Configuration Server" II .....   | 92  |
| Figura 53 | Iteración 10 - Diagrama de clases del "Configuration Server" I .....   | 93  |
| Figura 54 | Iteración 10 - Diagrama de clases del "Configuration Server" II .....  | 94  |
| Figura 55 | Iteración 10 - Diagrama de clases del "Configuration Server" III ..... | 95  |
| Figura 56 | Iteración 10 - Diagrama de clases del "Configuration Server" IV .....  | 96  |
| Figura 57 | Iteración 10 - Diagrama de clases del "Configuration Server" V .....   | 97  |
| Figura 58 | Iteración 11 - Diagrama de clases del "Configuration Client".....      | 98  |
| Figura 59 | Iteración 11 - Diagrama de clases del "Configuration Server" I .....   | 99  |
| Figura 60 | Iteración 11 - Diagrama de clases del "Configuration Server" II .....  | 100 |
| Figura 61 | Iteración 11 - Diagrama de clases del "Configuration Server" III ..... | 101 |
| Figura 62 | Lista de clases de pruebas I - Configuration Server .....              | 105 |
| Figura 63 | Lista de clases de pruebas II - Configuration Server .....             | 106 |
| Figura 64 | Cobertura de código - Configuration Server .....                       | 107 |
| Figura 65 | Lista de clases de pruebas - Configuration Client.....                 | 107 |
| Figura 66 | Cobertura de código - Configuration Client.....                        | 107 |
| Figura 67 | Configuración del "Configuration Server" .....                         | 108 |
| Figura 68 | Configuración del repositorio del "Configuration Server" .....         | 109 |
| Figura 69 | Utilización de la anotación @Value .....                               | 109 |
| Figura 70 | Métricas sonarqube del framework propuesto.....                        | 115 |
| Figura 71 | Calificación SQALE.....                                                | 116 |
| Figura 72 | Pirámide de deuda técnica y coste para reducir el riesgo.....          | 116 |

|           |                                                                   |     |
|-----------|-------------------------------------------------------------------|-----|
| Figura 73 | Histórico SQALE y estrategia ROI.....                             | 117 |
| Figura 74 | Métricas sonarqube - Mantenibilidad.....                          | 118 |
| Figura 75 | Métricas sonarqube - Fiabilidad .....                             | 119 |
| Figura 76 | Resultado JProfiler - Recursos.....                               | 120 |
| Figura 77 | Resultado JProfiler - APIs .....                                  | 121 |
| Figura 78 | Diagrama de barras del resultado de la encuesta por escalas ..... | 121 |
| Figura 79 | Evento - primer grupo.....                                        | 145 |
| Figura 80 | Evento - segundo grupo .....                                      | 145 |
| Figura 81 | Presentación - primer grupo I.....                                | 146 |
| Figura 82 | Presentación - primer grupo II.....                               | 146 |
| Figura 83 | Presentación - primer grupo III .....                             | 147 |
| Figura 84 | Presentación - primer grupo IV.....                               | 147 |
| Figura 85 | Presentación - primer grupo V.....                                | 147 |
| Figura 86 | Presentación - segundo grupo I .....                              | 148 |
| Figura 87 | Presentación - segundo grupo II .....                             | 148 |
| Figura 88 | Presentación - segundo grupo III .....                            | 149 |
| Figura 89 | Presentación - segundo grupo IV .....                             | 149 |
| Figura 90 | Dependencias del proyecto multi modulo .....                      | 150 |
| Figura 91 | Clase principal del módulo micro client .....                     | 151 |
| Figura 92 | Clase principal del módulo micro server.....                      | 151 |
| Figura 93 | Clase para configurar cors .....                                  | 152 |
| Figura 94 | Archivo readme.md del config manager - server I .....             | 153 |
| Figura 95 | Archivo readme.md del config manager - server II .....            | 154 |
| Figura 96 | Archivo readme.md del config manager - client.....                | 155 |
| Figura 97 | Colección de APIs - postman .....                                 | 156 |

## Resumen

La administración de la configuración de las aplicaciones de software es fundamental en las fases de desarrollo, implementación y despliegue de las mismas. Las configuraciones suelen referirse a servicios externos, como conexiones a bases de datos, credenciales encriptadas, sistemas de correo electrónico, plataformas de mensajería, entre otros. Estas configuraciones tienden a variar según el entorno de despliegue de las aplicaciones. El presente estudio de investigación plantea la creación del marco de trabajo denominado “config-manager” que posibilite la externalización de la configuración de aplicaciones multitenant. En una aplicación multitenant, la misma instancia de la aplicación funciona para varios clientes, conocidos como tenants, que comparten datos comunes y a la vez cuentan con datos aislados entre sí. Se propone establecer un punto central para la gestión de dichas configuraciones, otorgando a la aplicación la habilidad de operar en diversos entornos y atender a varios tenants sin necesidad de modificar o recompilar el código. Con el fin de evaluar la contribución del marco propuesto, se llevó a cabo un taller dirigido con profesionales del desarrollo de software, seguido de una encuesta de satisfacción de usuarios posterior al uso, donde se obtuvo una puntuación cercana a 1 que indica que se ha obtenido una mayor satisfacción en los participantes, adicionalmente se evaluaron otras métricas para conocer el estado de los atributos de calidad del software, como la eficiencia, fiabilidad y mantenibilidad donde se obtuvo como resultado un bajo consumo de recursos, cero incidencias con alta cobertura de código y deuda técnica igual a cero respectivamente.

Palabras Clave: Externalización de la Configuración, Entorno Multitenant, Framework.

## **Abstract**

Configuration management of software applications is critical in the development, implementation, and deployment phases of the application. Configurations typically refer to external services, such as database connections, encrypted credentials, email systems, messaging platforms. These configurations tend to vary depending on the application deployment environment. This research study proposes the creation of the framework called “config-manager” that enables the externalization of multitenant application configuration. In a multitenant application, the same instance of the applications server’s multiple clients, known as tenants, who share common data and at the same time have data isolated from each other. It is proposed to establish a central point for the management of such configurations, giving the application the ability to operate in various environments and serve multiple tenants without the need to modify or recompile the code. To evaluate the contribution of the proposed framework, a workshop was conducted with software development professionals, followed by a post-use user satisfaction survey, where a score close to one was obtained, indicating that greater satisfaction was obtained among the participants. Additionally, other metrics were evaluated to know the status of the software quality attributes, such as efficiency, reliability, and maintainability, where the results were low resource consumption, zero incidents with high code coverage and technical debt equal to zero, respectively.

**Keywords:** Externalize the Configuration, Multitenant Environment, Framework.

## CAPITULO 1: INTRODUCCION

### 1.1. Situación Problemática

La externalización de la configuración, que consiste en trasladar la información de configuración del artefacto de despliegue de una aplicación hacia una ubicación centralizada, se considera una práctica recomendada en el ámbito de DevOps (Balalaie et al., 2016). El objetivo principal de este enfoque es simplificar la gestión y supervisión de los datos de configuración entre las aplicaciones y sus instancias. Este enfoque constituye el fundamento del patrón de diseño denominado Externalización de la Configuración (ckittel, 2023). No obstante, durante el proceso de desarrollo de software, el objetivo de lograr una aplicación multitenant, puede enfrentar desafíos que dificultan su cumplimiento. Uno de estos desafíos radica en la necesidad de que las aplicaciones sean altamente configurables, con ajustes específicos para cada tenant (Hall, 2010). Además, la implementación de un framework multitenant sólido, robusto y escalable desde cero representa un desafío considerable. Esto se debe a la complejidad que implica para los desarrolladores comprender el manejo de las solicitudes web, cuando los usuarios pertenecen a un tenant en particular (Bien & Dan Thu, 2015).

Este suceso provoca que los equipos de desarrollo empleen varias herramientas, con el objetivo de reducir el tiempo de desarrollo invertido en sus aplicaciones. No obstante, dichas herramientas simplifican la labor de los desarrolladores al ocultar la complejidad, lo cual facilita la creación de aplicaciones sencillas. Sin embargo, dificultan la elaboración de aplicaciones más complejas que demandan un conocimiento detallado de la herramienta a un nivel más profundo. (Bien & Dan Thu, 2015).

En los proyectos, es prioritario utilizar diversas herramientas, ya que ninguna de ellas proporciona todas las funcionalidades requeridas para una aplicación multitenant. En ocasiones, estas herramientas ofrecen características que no están directamente relacionadas, lo que puede contravenir el principio de Responsabilidad Única (Kale & Borhade, 2013).

En resumen, para incluir el soporte multitenant en una aplicación, sin afectar sus atributos de calidad como la mantenibilidad, portabilidad, performance y compatibilidad, es importante, tener claras las características necesarias que una aplicación requiere para soportar múltiples tenants de forma transparente.

## **1.2. Formulación del Problema**

### **1.2.1. Problema General**

¿De qué manera se puede implementar el framework “config-manager” para externalizar la configuración en entornos multitenant?

### **1.2.2. Problemas Específicos**

¿De qué manera se puede implementar el framework “config-manager” con un alto nivel de mantenibilidad para externalizar la configuración en entornos multitenant?

¿De qué manera se puede implementar el framework “config-manager” con un alto nivel de fiabilidad para externalizar la configuración en entornos multitenant?

¿De qué manera se puede implementar el framework “config-manager” con un alto nivel de eficiencia para externalizar la configuración en entornos multitenant?

¿De qué manera se puede implementar el framework “config-manager” con un alto nivel de satisfacción del usuario para externalizar la configuración en entornos multitenant?

### 1.3. Justificación Teórica

En la actualidad, los equipos de desarrollo recurren a la externalización de la configuración de sus aplicaciones, con el propósito de garantizar su portabilidad y consistencia en diferentes entornos (desarrollo, pruebas, producción). Para lograr este objetivo, utilizan herramientas y servicios especializados (ckittel, 2023), Sin embargo, muchas de estas herramientas no brindan soporte a los diversos tipos de arquitectura, como en la arquitectura multitenant, donde la gestión de la configuración a nivel de tenant no se encuentra totalmente soportada.

Mas aun considerando que los atributos como la flexibilidad y agilidad de los servicios en la nube son posibles en gran medida a su arquitectura multitenant (Aditi & Parmeet, 2023).

En este sentido, el marco de trabajo propuesto busca fortalecer los aspectos de la calidad del software, facilitando una mayor flexibilidad en el mantenimiento y adaptación de las aplicaciones al externalizar la configuración, permitiendo una mejor gestión de los entornos de ejecución de aplicaciones, que requieren distintas configuraciones entre ellos y brindando soporte nativo para la configuración por tenant.

Los sistemas de gestión de configuración eficientes, escalables y robustos que permite a los clientes gestionar sus configuraciones de forma centralizada, externalizada y dinámica, son cada vez más necesario debido a la popularidad de la tendencia de implementar varios contenedores de numerosas aplicaciones distribuidas (Ji et al., 2024).

Este enfoque responde a la necesidad de sistemas más escalables y adaptables como el propuesto, donde se permiten modificaciones dinámicas de la configuración sin necesidad de recompilación.

#### **1.4. Justificación Practica**

El presente estudio de investigación plantea la creación de un marco de trabajo para externalizar la configuración de aplicaciones basadas en arquitecturas multitenant, y para cumplir con este propósito, se hace revisión de diversos artículos científicos relacionados a las buenas prácticas del desarrollo de software, patrones de diseño y herramientas de externalización, para determinar las características que debería tener una herramienta que satisfaga las necesidades y exigencias de la externalización de la configuración de las aplicaciones multitenant de manera sencilla y eficiente.

El framework propuesto se enfoca en agilizar la gestión de tenants, abstraer el uso de sus configuraciones en las aplicaciones clientes y en facilitar la integración con aplicaciones nuevas y/o adaptación con aplicaciones existentes.

Este enfoque pretende disminuir los tiempos de desarrollo y mantenimiento de las configuraciones realizadas por el equipo de desarrollo y soporte respectivamente, con el objetivo que se enfoquen en otras tareas más importantes.

## **1.5. Objetivos**

### **1.5.1. Objetivo General**

Implementar el framework “config-manager” para externalizar la configuración en entornos multitenant.

### **1.5.2. Objetivos Específicos**

Implementar el framework “config-manager” con un alto nivel de mantenibilidad para externalizar la configuración en entornos multitenant.

Implementar el framework “config-manager” con un alto nivel de fiabilidad para externalizar la configuración en entornos multitenant.

Implementar el framework “config-manager” con un alto nivel de eficiencia para externalizar la configuración en entornos multitenant

Implementar el framework “config-manager” con un alto nivel de satisfacción del usuario para externalizar la configuración en entornos multitenant.

## **CAPITULO 2 MARCO TEORICO**

### **2.1. Marco Filosófico o Epistemológico de la Investigación**

La epistemología y la investigación cuantitativa son términos frecuentemente utilizados en el campo de la ciencia, y su campo de estudio se enmarca en la literatura científica, lo que lo eleva en su estructura de estudio y campo de aplicación, sin embargo, la investigación cuantitativa constituye una dimensión clave que conjuga con la investigación científica (Yucra Quispe & Bernedo Villalta, 2020).

El presente trabajo de investigación sigue un enfoque cuantitativo, sustentado en la epistemología positivista, que busca explicaciones objetivas, cuantificables y medibles de la realidad, el cual permite la formulación de hipótesis que pueden ser verificadas (Huamán Rojas et al., 2022) a través de las métricas obtenidas en la presente investigación.

### **2.2. Antecedentes de Investigación**

Se ha realizado una profunda investigación de diversos artículos científicos que hacen referencia a herramientas y/o propuestas de herramientas que permiten implementar y gestionar de forma eficiente la configuración de tenants en las aplicaciones.

**2.2.1** Tradicionalmente, las bases de datos relacionales han sido utilizadas para almacenar los datos de los tenants por aplicaciones de esta arquitectura, almacenando los datos en las mismas tablas, generando problemas al extender o modificar un esquema existente.

Debido a esta problemática, los autores Sharma & Kaur (2021) proponen un enfoque donde utilizan una base de datos orientada a documentos en vez de una tradicional, con el objetivo de beneficiarse de la flexibilidad del diseño de esquemas y la característica de fragmentación, para implementar un

almacenamiento eficiente de datos de los tenants que estas bases de datos ofrecen.

Se propone la implementación de una aplicación multitenant usando MongoDB por ser la base de datos basada en documentos más popular, específicamente, se hace uso de dos características de la base de datos para satisfacer las necesidades de una aplicación multitenant:

- No se necesita definir el esquema de la base de datos antes de almacenar los registros, lo que permite que los tenants tengan distintos atributos.
- La facilidad para distribuir los datos en múltiples fragmentos, denominados shards. Aseguran el aislamiento de los datos de los tenants.

El algoritmo de 8 pasos que resume la creación de la base de datos multitenant se observa a continuación en la Figura 1.

### Figura 1

#### *Algoritmo para la creación de la base de datos multitenant*

1. For each tenant *i* do
2.     Assign an identifier, *tenant\_id* to the tenant *i*
3.     Collect tenant data
4.     Enter the data into collection *c* along with *tenant\_id*
5. end for
6. Select shard key and type of sharding to be performed
7. Create shards for tenants
8. Allocate tenant data to appropriate shard.

*Nota.* Tomado de “Implementing a Multitenant System using a Document-based NoSQL Database”, por Sharma & Kaur, 2021.

Como resultado de la investigación, se constató, que la base de datos MongoDB permitió el aislamiento de los tenants en la misma base de datos mediante la fragmentación, la que a su vez aumentó el rendimiento y escalabilidad de la aplicación.

**2.2.2** Ouh y Kok Siew Gan (2023) realizaron un estudio exploratorio del estilo arquitectónico multitenant, revisando diversos trabajos existentes con el objetivo de entender las propiedades necesarias de dicha arquitectura, y proponer una guía para soportar arquitecturas multitenant sobre tecnología nueva o existente.

Como primer paso, los autores revisaron la literatura existente del diseño multitenant, obteniendo como resultado las propiedades deseadas en la Tabla 1.

**Tabla 1**

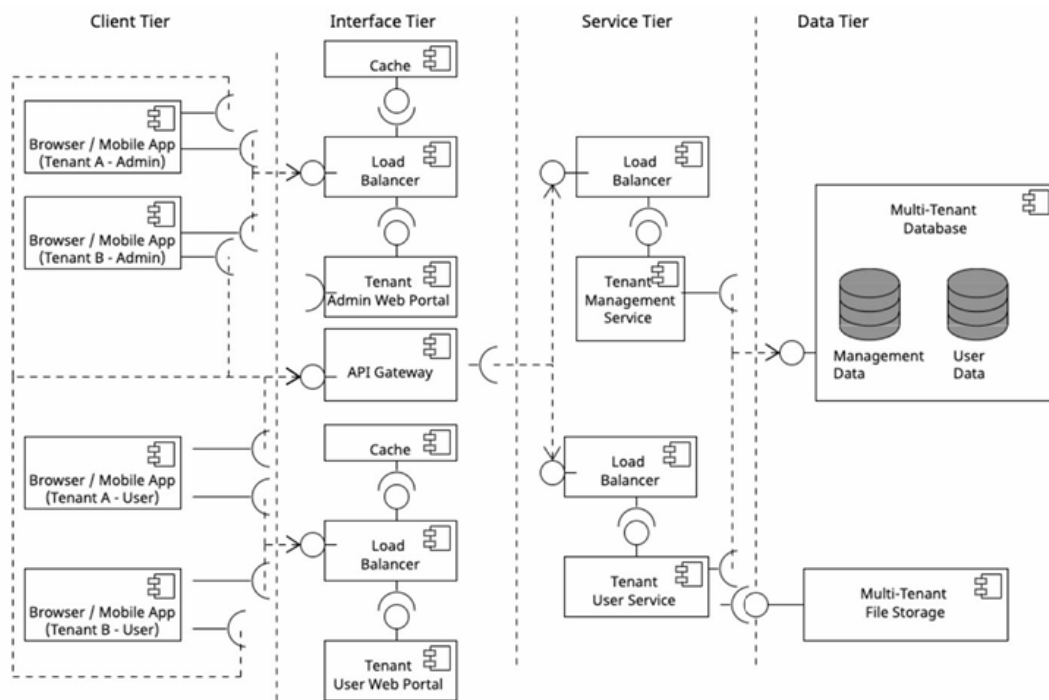
*Propiedades deseadas de la Arquitectura.*

| Atributo de Calidad       | Propiedad                  |
|---------------------------|----------------------------|
| Adecuación funcional      | Afinidad                   |
| Usabilidad                | Altamente configurable     |
| Mantenibilidad            | Intercambio de recursos    |
| Eficiencia de rendimiento | Elasticidad                |
| Fiabilidad                | Alta disponibilidad        |
| Seguridad                 | Aislamiento de recursos    |
| Compatibilidad            | Personalización            |
| Portabilidad              | Gestión de la variabilidad |

*Nota.* En la tabla se muestra la relación del atributo de Calidad con la propiedad deseada. Adaptado de “An Exploratory Study of Architectural Style and Effort Estimation for Multi-Tenant Microservices-Based Software as Service (SaaS)”, por Ouh & Kok Siew Gan, 2023.

Las propiedades deseadas se encuentran alineadas a los 8 atributos de calidad de la ISO25010.

Luego procedieron a diagramar el estilo propuesto, haciendo uso de notaciones UML, como se visualiza en la Figura 2.

**Figura 2***Vista funcional del estilo propuesto*

*Nota.* Tomado de "An Exploratory Study of Architectural Style and Effort Estimation for Multi-Tenant Microservices-Based Software as Service (SaaS)", por Ouh & Kok Siew Gan, 2023.

Finalmente aplicaron el estilo propuesto en un caso de estudio, donde evolucionaron una aplicación existente de un único tenant para admitir múltiples tenants, todo ello para registrar su esfuerzo en un desarrollo real y validarlo contra un modelo de predicción de esfuerzo web existente.

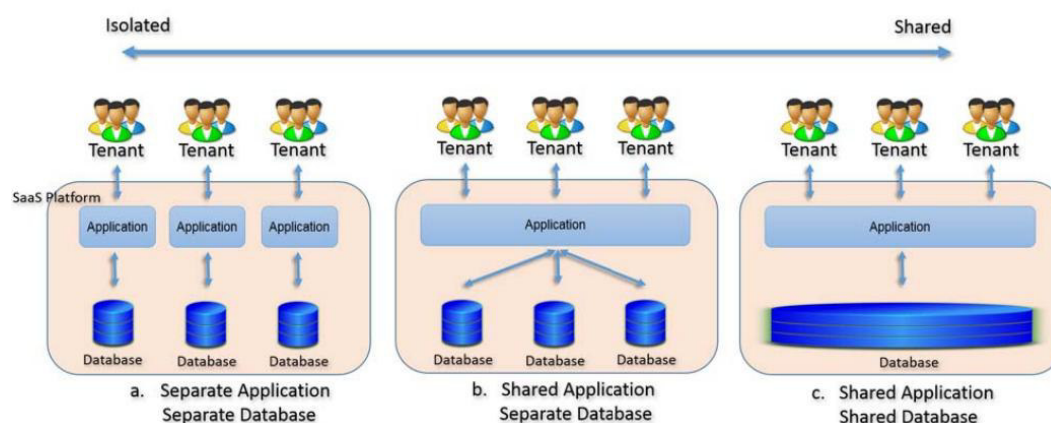
El resultado proporciona un enfoque para que los proveedores de servicios puedan usar un modelo de predicción validado y obtener una estimación temprana del esfuerzo requerido por proyectos web que desean evolucionar a una arquitectura multitenant.

**2.2.3** Karatas et al (2017) investigaron la estrategia de almacenamiento más comúnmente empleada en arquitecturas multitenant y los parámetros utilizados para su selección. Para alcanzar este propósito, llevaron a cabo una revisión exhaustiva de numerosos estudios provenientes de diversas fuentes indexadas.

En una primera etapa, en la Figura 3 los autores exponen las estrategias de almacenamiento de datos utilizadas para definir el alcance:

**Figura 3**

*Estrategias de almacenamiento de datos*



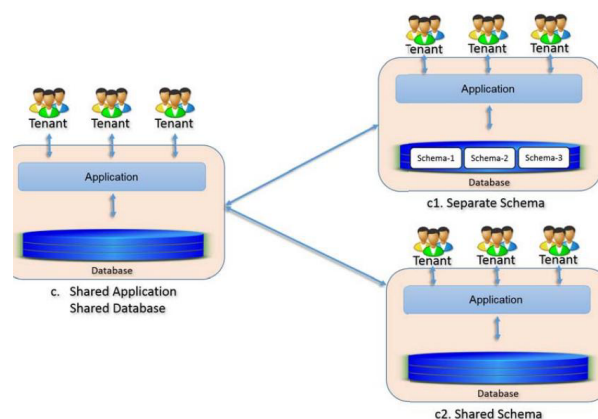
*Nota.* Tomado de “Multi-tenant architectures in the cloud: A systematic mapping study”, por Karatas et al., 2017.

- A. Base de datos y Aplicación separadas,** cada tenant es dueño de su propia instancia de aplicación y base de datos.
- B. Base de datos separada y Aplicación compartida,** todos los tenants usan la misma instancia de la aplicación y son dueños de sus bases de datos.
- C. Base de datos y Aplicación compartidas,** todos los tenants usan la misma instancia de la aplicación y base de datos, esta estrategia a su vez se divide en dos modelos para el manejo de la base de datos como se observa en la Figura 4.

- a. **Esquema separado**, existe un esquema para cada tenant en la misma base de datos.
- b. **Esquema compartido**, el mismo esquema es usado para todos los tenants.

**Figura 4**

*Esquemas separados y compartidos*



*Nota.* Tomado de “Multi-tenant architectures in the cloud: A systematic mapping study”, por Karatas et al., 2017.

En la segunda etapa, los investigadores establecen su metodología de investigación al identificar las siguientes preguntas:

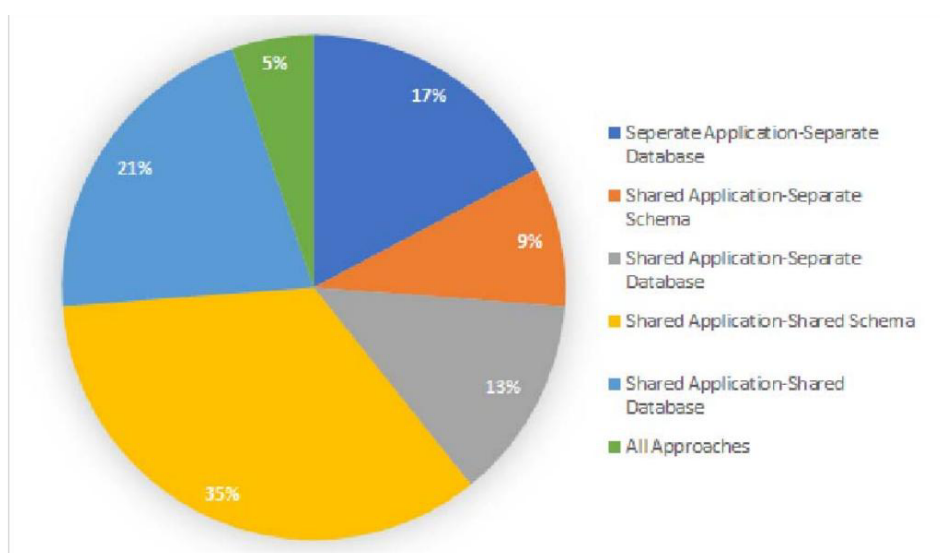
- ¿Qué tipos de publicaciones han sido hechas acerca del multitenancy?
- ¿Qué clases de estudios han sido producidos incluyendo años y fuentes?
- ¿Qué estrategia de almacenamiento es preferido para el multitenancy?
- ¿Qué criterios han sido tomados en cuenta para determinar la estrategia de almacenamiento preferida?
- ¿Cuál es el título más buscado para sistemas multitenant y cuál es el propósito de la investigación?
- ¿Cuáles son los beneficios conocidos y limitaciones de los sistemas multitenant?

y algunas palabras reservadas con el objetivo de obtener los artículos más relevantes relacionados a la investigación.

Como resultado de las publicaciones examinadas, los autores generaron distribuciones usando diversas dimensiones, donde se destaca la Figura 5 que determina que la estrategia de almacenamiento preferida es la **Aplicación y esquema compartido** con un 35% de uso.

**Figura 5**

*Estrategia de almacenamiento preferida*

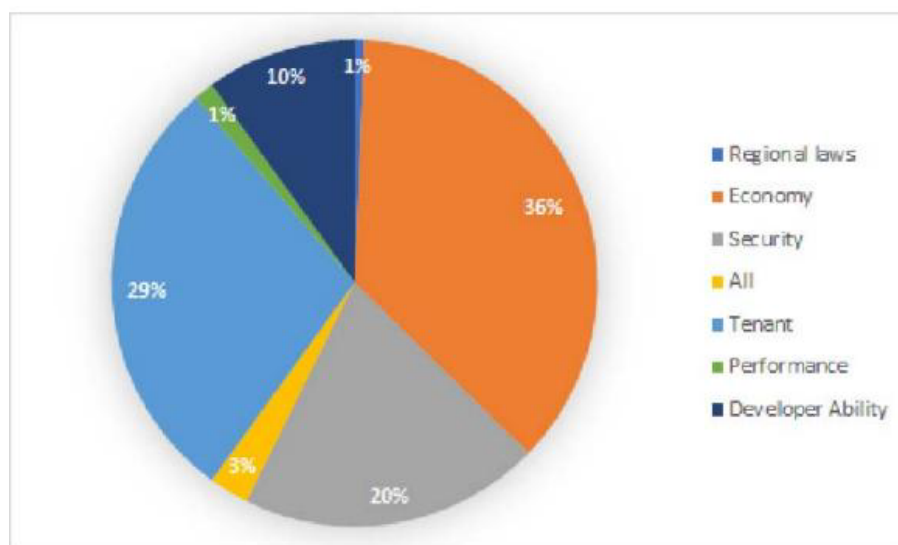


*Nota.* Tomado de “Multi-tenant architectures in the cloud: A systematic mapping study”, por Karatas et al., 2017.

Y profundizando un poco más también determinan cual es el criterio que predomina para la elección de una estrategia de almacenamiento, como se muestra en la Figura 6.

**Figura 6**

*Criterio para determinar la estrategia de almacenamiento preferida*



*Nota.* Tomado de “Multi-tenant architectures in the cloud: A systematic mapping study”, por Karatas et al., 2017.

Finalmente, los autores concluyen que los sistemas multitenancy, solo son aplicados en un número limitado de áreas, lo que quiere decir que falta una mayor expansión y que a su vez la infraestructura de software tienes que prepararse para facilitar su implementación.

**2.2.4** Odun-Ayo et al. (2017) presentan un estudio exhaustivo de la literatura existente relacionada al multitenancy. Donde los autores revisaron diversos trabajos de investigación relacionadas a las aplicaciones multitenant y las tendencias de estas en la nube, con el propósito de describir las características esenciales que deben considerar y la arquitectura de estas.

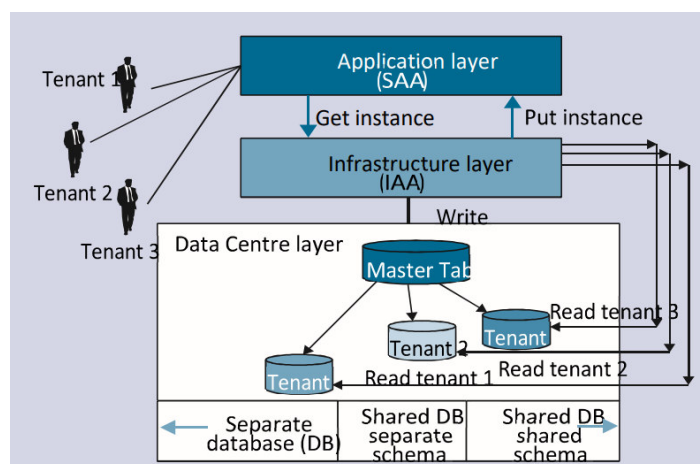
Del estudio, se desprenden las **Características Multitenant** que de acuerdo con la literatura son esenciales en una aplicación multitenant.

- Compartir recursos de hardware.
- Alto grado de configurabilidad.
- Aplicación y base de datos compartidas.

La **Gestión de Datos** es un punto muy importante ya que define la estrategia de almacenamiento de los datos de los tenants, de los que destacan 3 enfoques para gestionar esta data en la nube:

- Almacenar los datos de los tenants en bases de datos separadas.
- Almacenando los datos de todos los tenants en la misma base de datos, y cada tenant usando su propio esquema.
- Usando la misma base de datos para almacenar los datos de los tenants.

En la Figura 7 se muestra una arquitectura multitenant general en la nube, compuesta por 3 capas (Aplicación, Infraestructura y centro de datos).

**Figura 7***Arquitectura multitenant en la nube*

*Nota.* Tomado de "Cloud Multi-Tenancy: Issues and Developments", por Odun-Ayo et al., 2017.

Como mencionan los autores, los entornos en la nube brindan una escalabilidad prácticamente ilimitada que permite resolver el problema de escalamiento de las aplicaciones, pero a su vez plantea nuevos problemas de seguridad y privacidad debido a la transferencia de tareas de procesamiento y almacenamiento a los proveedores cloud.

Del estudio se concluye que las aplicaciones multitenant usualmente comparten la misma aplicación y esquema como estrategia de almacenamiento de datos y la razón predominante son temas económicos.

**2.2.5** Los autores Sayagh et al. (2017) realizaron un estudio empírico, en el cual analizaron más de 2000 proyectos java alojados en GitHub, con el objetivo de analizar la popularidad de los frameworks de configuración y las características que poseen.

El estudio se enfoca en el propósito general de la configuración de los 11 frameworks de programación listados en la Tabla 2.

**Tabla 2**

*Frameworks de configuración estudiados y sus firmas*

| #  | Framework   | Released | Signature                                 |
|----|-------------|----------|-------------------------------------------|
| 1  | Properties  | 01.1996  | Import java.util.Properties               |
| 2  | System      | 01.1996  | System.gertProperty                       |
| 3  | jConfig     | 10.2001  | Import org.jconfig.*                      |
| 4  | Preferences | 07.2003  | Import java.util.prefs.*                  |
| 5  | Spring      | 06.2003  | Import org.springframework.*              |
| 6  | Commons     | 11.2005  | Import org.apache.commons.configuration.* |
| 7  | Constretto  | 05.2010  | import org.constretto.*                   |
| 8  | Typesafe    | 12.2011  | import com.typesafe.config                |
| 9  | Deltaspike  | 10.2012  | Import org.apache.deltaspike.*            |
| 10 | Owner       | 12.2012  | Import org.aeonbits.owner.*               |
| 11 | CFG4J       | 07.2015  | Import org.cfg4j.*                        |

*Nota.* Tomado de “Does the Choice of Configuration Framework Matter for Developers? Empirical Study on 11 Java Configuration Frameworks”, por Sayagh et al., 2017.

Con el objetivo de entender la diferencia entre los frameworks de configuración anteriormente listados, los autores construyeron la taxonomía de estos, basados en sus características y propiedades, para ello estudiaron la documentación publica, blogs, fórums asociados a cada framework de forma manual, obteniéndose como resultado 17 propiedades agrupadas en 3 dimensiones.

En la Tabla 3 se observa que los frameworks Common y Spring satisfacen la mayoría de las características de un framework de configuración, pero este indicador no garantiza que sean los más utilizados.

**Tabla 3***Taxonomía de los 11 frameworks de configuración*

| Dimensions          | Properties                         | 1    | 2    | 3   | 4    | 5     | 6     | 7    | 8    | 9    | 10   | 11  |
|---------------------|------------------------------------|------|------|-----|------|-------|-------|------|------|------|------|-----|
| General Properties  | Universal                          | ✓    | ✓    | ✓   | ✓    |       | ✓     | ✓    | ✓    |      | ✓    | ✓   |
|                     | Part of SDK                        | ✓    | ✓    |     | ✓    |       |       |      |      |      |      |     |
|                     | Age                                | high | high | med | med  | med   | med   | low  | low  | low  | low  | low |
|                     | Quality of Documentation           | high | high | low | high | high  | high  | med  | high | med  | high | med |
|                     | Actively Maintained                | ✓    | ✓    |     | ✓    | ✓     | ✓     |      | ✓    | ✓    | ✓    | ✓   |
| Feature Richness    | Multiple Storage Formats           | low  | low  | med | med  | med   | high  | high | med  | med  | low  | med |
|                     | Hierarchical Configuration Struct. |      |      | ✓   |      | ✓     | ✓     | ✓    | ✓    | ✓    | ✓    |     |
|                     | Hierarchical Overriding            |      |      |     |      | ✓     | ✓     | ✓    | ✓    | ✓    | ✓    |     |
|                     | Multiple Data Sources              |      |      | ✓   | ✓    | ✓     | ✓     |      | ✓    |      |      | ✓   |
|                     | Variable Substitution              |      |      |     |      | ✓     | ✓     |      | ✓    | ✓    | ✓    |     |
|                     | #API methods                       | 15   | 5    | 273 | 39   | 1,681 | 1,022 | 191  | 109  | 44   | 37   | 57  |
|                     | #Annotations                       | 0    | 0    | 0   | 0    | 242   | 0     | 6    | 1    | 16   | 12   | 0   |
| Programming Support | Dependencies                       | none | none | med | none | high  | high  | med  | none | none | none | med |
|                     | Distributed Environment Support    |      |      | ✓   |      | ✓     | ✓     |      |      |      |      | ✓   |
|                     | Type-safety                        |      |      | ✓   | ✓    | ✓     | ✓     | ✓    | ✓    | ✓    | ✓    | ✓   |
|                     | Notification Mechanisms            |      |      | ✓   |      | ✓     | ✓     |      |      |      | ✓    |     |
|                     | Configuration Injection            |      |      |     |      | ✓     | ✓     | ✓    |      | ✓    | ✓    |     |

*Nota.* Tomado de “Does the Choice of Configuration Framework Matter for Developers?

Empirical Study on 11 Java Configuration Frameworks”, por Sayagh et al., 2017.

Adicionalmente como resultado de la revisión de todos los repositorios de proyectos java, obtuvieron un índice de popularidad entre los frameworks. Y como se observa en la Tabla 4 hay un gran favoritismo sobre la no utilización de un framework especializado, como se muestra en las dos primeras posiciones.

**Tabla 4***Proyectos usando solo el framework.*

| Framework            | # Projects   | % Projects | # 1 CF     |
|----------------------|--------------|------------|------------|
| System               | 821          | 42.36      | 347        |
| Properties           | 600          | 30.96      | 142        |
| Spring               | 91           | 4.70       | 38         |
| Preferences          | 57           | 2.94       | 6          |
| Commons              | 37           | 1.91       | 5          |
| Typesafe             | 14           | 0.72       | 4          |
| Deltaspike           | 3            | 0.15       | 0          |
| CFG4J                | 1            | 0.05       | 0          |
| Constretto           | 1            | 0.05       | 0          |
| jConfig              | 0            | 0          | 0          |
| Owner                | 0            | 0          | 0          |
| <b>Any Framework</b> | <b>1,034</b> | <b>-</b>   | <b>542</b> |

*Nota.* Tomado de “Does the Choice of Configuration Framework Matter for Developers?

Empirical Study on 11 Java Configuration Frameworks”, por Sayagh et al., 2017.

**2.2.6** Los proveedores de nube están optando por arquitecturas multitenant con el objetivo de aumentar su eficiencia para el manejo de grandes volúmenes de datos, que últimamente se están volviendo más complejos, debido al aumento de las dimensiones de los datos, su falta de estructura y mayor necesidad de espacio de almacenamiento.

Por este motivo los autores Sharma and Kaur (2019) proponen una arquitectura de base de datos multitenant usando almacenamiento de datos NoSQL.

Como resultado de un mapeo sistemático, los autores encontraron que la estrategia de almacenamiento de base de datos y aplicación compartida es la estrategia más popular en entornos multitenant y para comprobar la factibilidad de su propuesta, los autores la aplicaron a un caso de estudio convirtiendo una aplicación de un solo tenant a una multitenant.

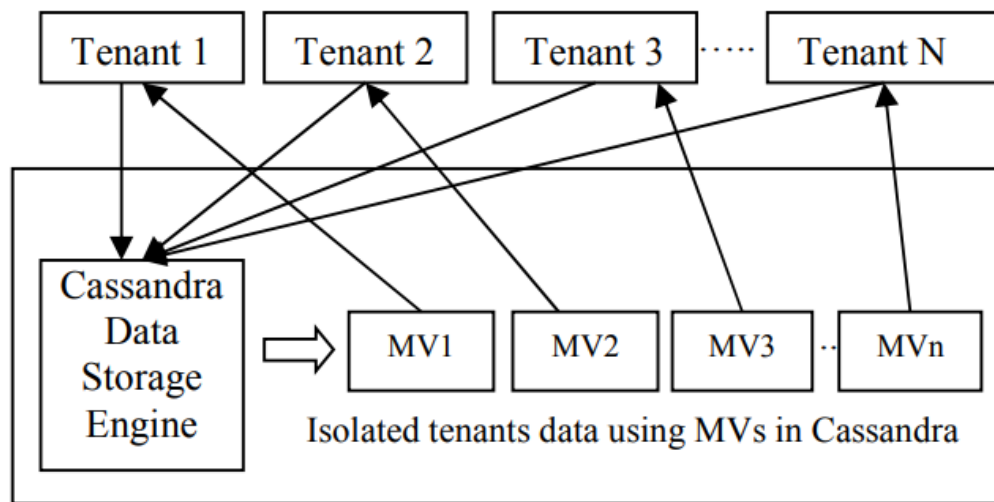
Para lograr esto, usaron la base de datos Cassandra y siguieron los pasos a continuación:

1. Extracción de la data existente y diseño del esquema de la tabla.
2. Importación de los datos extraídos a la base de datos Cassandra.
3. Aislamiento de los tenants creando vistas materializadas.

Como se observa en la Figura 8, los tenants obtienen un aislamiento a través de las vistas materializadas mientras permite la flexibilidad en el diseño o extensión del esquema.

**Figura 8**

*Arquitectura del sistema propuesto*



*Nota.* Tomado de “A Multitenant Data Store Using a Column Based NoSQL Database”, por Sharma & Kaur, 2019.

Los resultados obtenidos mostraron que la arquitectura multitenant generó los valores deseados en menos tiempo para grandes almacenes de datos, pero no es sugerido para aplicaciones que cuentan con pequeños conjuntos de datos.

**2.2.7** Los autores Lee et al (2020) en su trabajo de investigación implementaron una API GraphQL para solucionar un problema con la aplicación HEMS (Sistema de gestión de energía para el hogar), la cual tenía que realizar muchas peticiones a una aplicación REST heredada para obtener la información requerida.

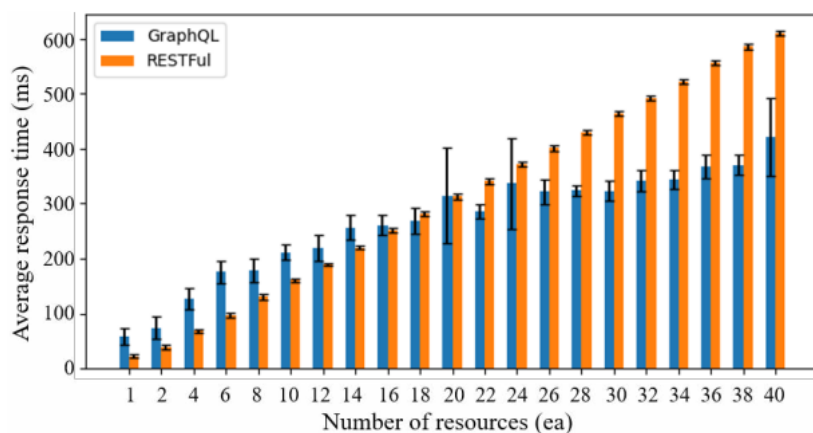
Optaron por esta tecnología, porque los clientes GraphQL pueden solicitar los datos deseados al servidor, seleccionando los tipos de objetos y campos explícitamente. Esta característica encaja perfectamente con el problema a solucionar, ya que la aplicación HEMS, permite configurar paneles de control, que solo muestran la información que los usuarios desean visualizar.

Con el objetivo de validar la mejora en la aplicación, los autores realizaron un experimento para comparar los tiempos de respuesta y el tamaño de las respuestas de las peticiones usando la nueva API GraphQL y la API REST existente.

Como se observa en la Figura 9, cuando el número de recursos es menor a 20, el promedio de tiempo de respuesta de REST es más lento que el de GraphQL, y cuando es mayor a 20 la situación cambia.

**Figura 9**

*Gráfico comparativo de los tiempos de respuesta*

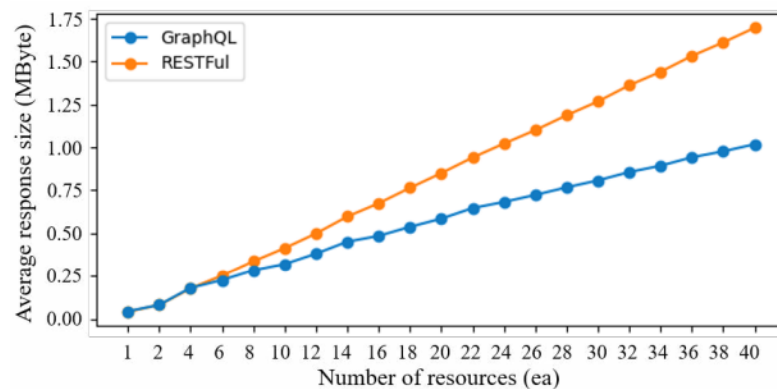


*Nota.* Tomado de “Performance Measurement of GraphQL API in Home ESS Data Server”, por Lee et al., 2020.

Además, se observa en la Figura 10 que el tamaño de la respuesta se incrementa proporcionalmente a los recursos solicitado a la vez para la aplicación RESTful a diferencia de GraphQL donde la respuesta tiene menor tamaño.

**Figura 10**

*Gráfico comparativo de los tamaños de respuesta*



*Nota.* Tomado de “Performance Measurement of GraphQL API in Home ESS Data Server”, por Lee et al., 2020.

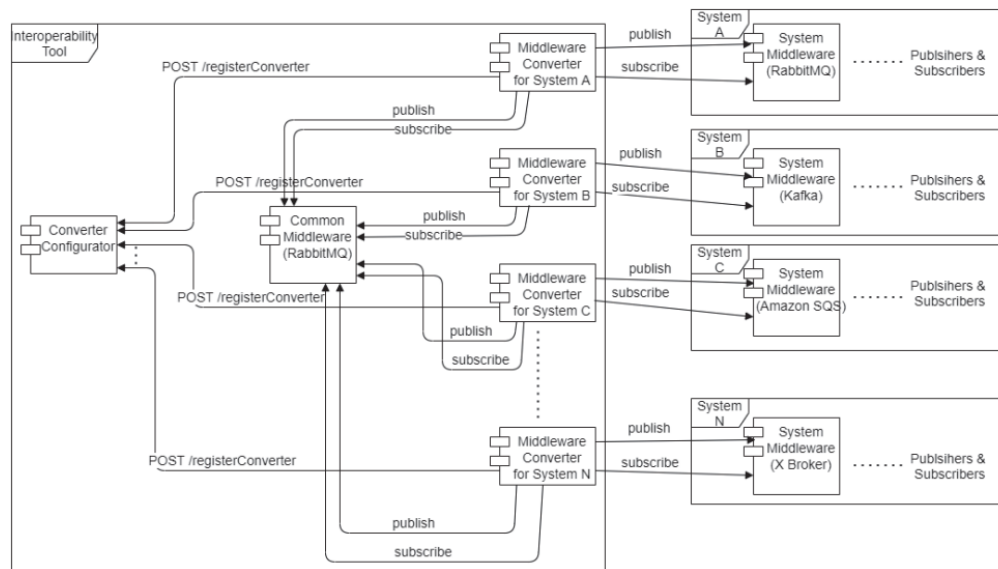
Como resultado, se concluyó que es mejor usar REST para servicios simples con menos recursos solicitados a la vez, y GraphQL es mejor para servicios complejos con gran cantidad de recursos solicitados a la vez.

**2.2.8** Con el objetivo de proveer interoperabilidad entre sistemas de diferentes arquitecturas de microservicios, el autor Bayramcavus et al. (2021) presenta una propuesta solución a través de una puerta de enlace que está diseñada para trabajar bajo el sistema de publicación/suscripción para permitir la comunicación entre sistemas.

La herramienta propuesta de interoperabilidad consiste principalmente en tres partes como se detalla en la Figura 11.

**Figura 11**

*Arquitectura de la herramienta de interoperabilidad*



*Nota.* Tomado de “Interoperability of Microservice-Based Systems”, por Bayramcavus et al., 2021.

- El **Converter Configurator** permite a los usuarios configurar que topics de que sistemas deben ser reenviados a que topics de que sistema.
- El **Middleware Converter** es el responsable de gestionar el reenvío de mensajes entre el *common middleware* y el *system middleware*.
- El **Common Middleware** es usado por el *middleware converter* para reenviar los mensajes entre diferentes sistemas.

A través de la propuesta solución, los autores desean habilitar la interoperabilidad de sistemas que utilizan diferentes middlewares de comunicación, reenviando los mensajes entre los topics de los sistemas. Como fase inicial la herramienta solo integra los middlewares RabbitMQ y Kafka, pudiendo ser extendido con más middlewares.

Como resultado de diversas pruebas, se pudieron obtener los tiempos de retraso de los mensajes al ser retransmitidos por la herramienta de interoperabilidad como se visualiza en la Figura 12.

**Figura 12**

*Tiempo de retraso entre componentes en milisegundos*

| Source  | Dest    | Messages per second from each publisher in milliseconds |      |      |       |       |
|---------|---------|---------------------------------------------------------|------|------|-------|-------|
|         |         | 40                                                      | 100  | 200  | 400   | 500   |
| A Pub   | A Sub   | 1.42                                                    | 1.30 | 1.10 | 1.98  | 2.88  |
| A Pub   | Conv. A | 1.38                                                    | 1.39 | 1.16 | 2.38  | 4.24  |
| Conv. A | Conv. B | 1.86                                                    | 1.35 | 1.23 | 4.04  | 6.04  |
| Conv. B | B Sub   | 2.94                                                    | 2.19 | 2.25 | 7.72  | 8.11  |
| A Pub   | B Sub   | 6.18                                                    | 4.93 | 4.64 | 14.14 | 18.39 |
| B Pub   | B Sub   | 2.70                                                    | 2.31 | 2.14 | 6.42  | 6.64  |
| B Pub   | Conv. B | 2.98                                                    | 2.36 | 2.63 | 21.04 | 22.95 |
| Conv. B | Conv. A | 2.24                                                    | 1.47 | 1.48 | 4.85  | 6.06  |
| Conv. A | A Sub   | 1.18                                                    | 1.17 | 1.00 | 2.54  | 3.86  |
| B Pub   | A Sub   | 6.40                                                    | 5.00 | 5.11 | 28.43 | 32.87 |

*Nota.* Tomado de “Interoperability of Microservice-Based Systems”, por Bayramcavus et al., 2021.

Se valida que la herramienta puede reenviar miles de mensajes entre diferentes sistemas con retraso menor a un segundo. También notan que el rendimiento de RabbitMQ es mejor que KafKa solo en su configuración básica.

**2.2.9** Debido al rápido desarrollo de las bases de datos NoSQL y sus tecnologías relacionadas en diversos escenarios, los autores Tang & Fan, (2016) en su trabajo de investigación evalúan el rendimiento de cinco bases de datos NoSQL (Redis, MongoDB, CouchBase, Cassandra and HBase) utilizando la herramienta de medición llamada YCSB (Yahoo Cloud Serving Benchmark), con el objetivo de que los usuarios puedan elegir fácilmente la base de datos que se adecue a sus requerimientos.

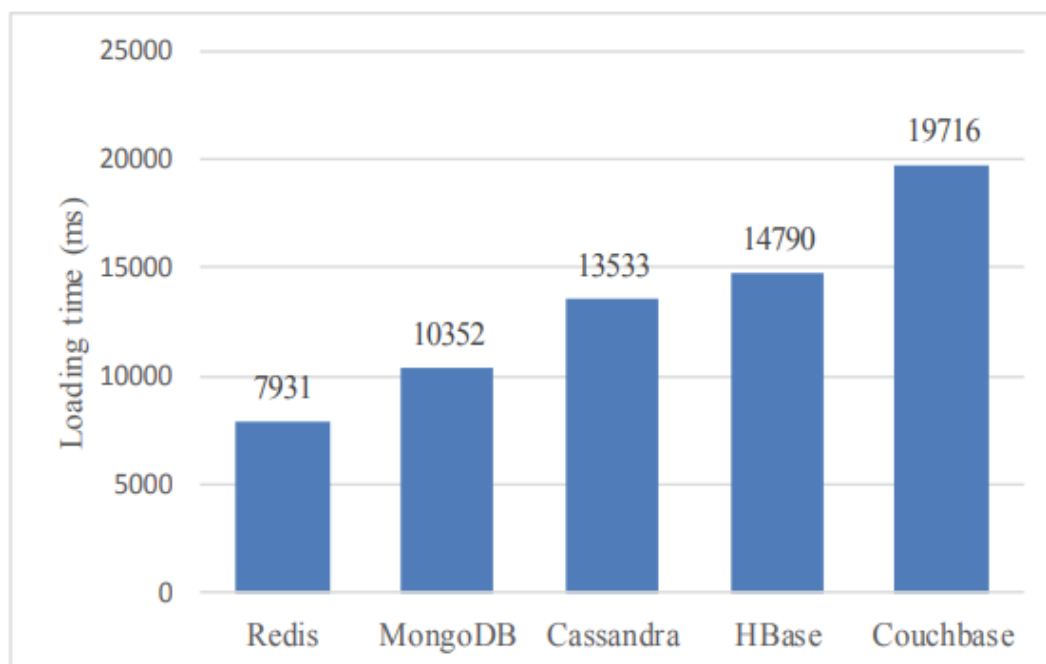
Los autores consideraron diferentes escenarios en sus experimentos, partiendo desde la inserción de registros hasta la ejecución de diferentes operaciones, como lecturas, actualizaciones y eliminaciones.

Estas cargas de trabajo fueron aplicadas en diferentes escenarios donde el tamaño de la data era incrementado gradualmente en todas las bases de datos.

Como resultado de los experimentos, se obtuvieron la Figura 13 y Figura 14, Redis tiene un mejor rendimiento comparado con las otras bases de datos NoSQL.

**Figura 13**

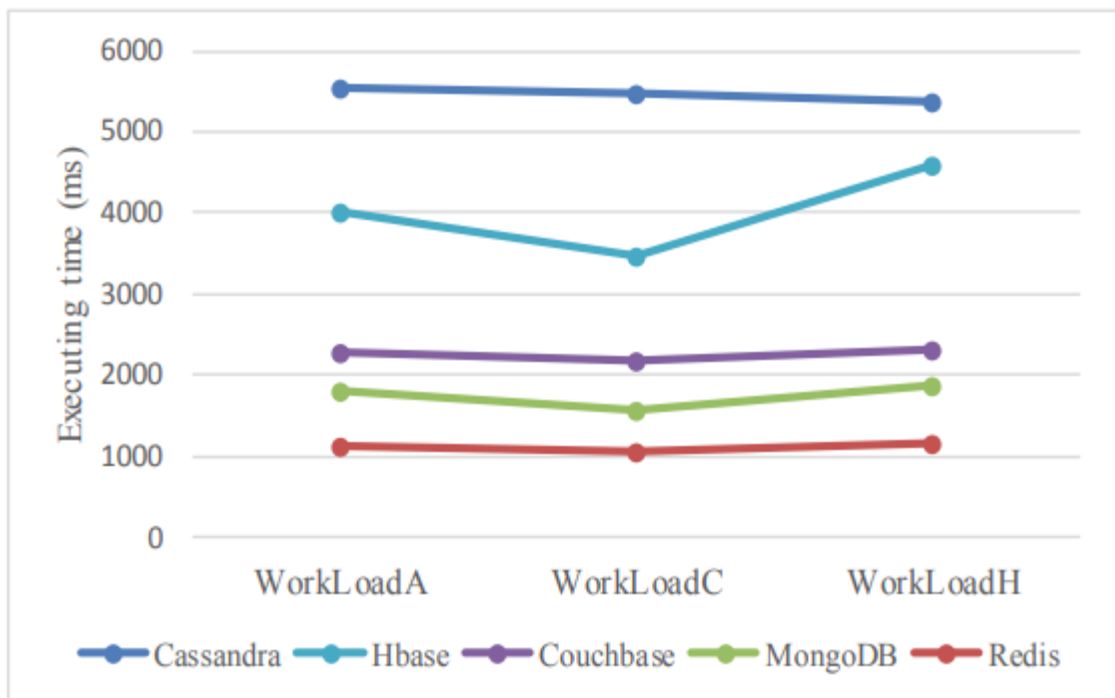
*Tiempo de carga para 100000 registros en 5 base de datos*



*Nota.* Tomado de “Performance Comparison between Five NoSQL Databases”, por Tang & Fan, 2016.

**Figura 14**

*Tiempo de ejecución para 100000 registros en 5 bases de datos*



*Nota.* Tomado de “Performance Comparison between Five NoSQL Databases”, por Tang & Fan, 2016.

La investigación de los autores concluye con que Redis es especialmente adecuado para la carga de registros y ejecutar cargas de trabajo. Aunque muestra la mejor eficiencia, Redis también tiene su limitación cuando se entrena a datos extremadamente grandes. Las bases de datos de documentos, seguidas de las bases de datos de columnas y familias tienen un buen rendimiento promedio, ya que poseen tanto la eficiencia como la escalabilidad.

## 2.3. Bases teóricas

### 2.3.1. Técnicas de Externalización

El autor Hofmann et al., s. f. (2017) en el capítulo 8 de su libro, describe las características que los microservicios deben tener para facilitar la transición entre entornos sin realizar cambios en los artefactos de despliegue.

Para lograr esto describe 3 técnicas diferentes de realizar la configuración de las aplicaciones a través de los entornos.

#### 2.2.1.1. Variables de Entorno

Esta técnica obtiene la configuración desde fuera de la aplicación utilizando las variables entornos, del sistema operativo que se describen con parejas de clave/valor. La Figura 15 presenta un ejemplo de variables de entorno.

#### Figura 15

*Asignación de variables de entorno en un script de Linux*

```
#!/bin/bash  
export VARIABLE1="Value 1"  
export VARIABLE2="Value 2"
```

*Nota.* Tomado de *Microservices Best Practices for Java* (p. 120), por Hofmann et al., 2016.

#### Ventajas

- Fácil configuración, no necesita la instalación de herramientas adicionales para la asignación de las variables.
- Soporte nativo por parte de los lenguajes de programación para el consumo de las variables.

**Desventajas**

- En entornos heterogéneos se contará con diferentes scripts por sistema operativo.
- No cuenta con un sistema de versionamiento.
- Las variables de entornos son duplicadas en todos los nodos donde la aplicación se ejecuta.
- Falta de mecanismos de seguridad.

### 2.2.1.2. Archivos de configuración

La utilización de archivos de configuración, que luego son cargados por la aplicación, es otra alternativa para externalizar la configuración.

Para hacer uso de esta técnica se requiere que el archivo de configuración exista en cada entorno, con sus respectivos valores. La Figura 16 detalla el contenido de un archivo YAML de ejemplo.

**Figura 16**

*Ejemplo de configuración de archivos usando YAML*

```
environments:
  local:
    url: http://test.visa.com/payments
  uat:
    url: http://test.visa.com/payments
  prod:
    url: http://www.visa.com/payments
```

*Nota.* La figura muestra un ejemplo de un archivo YAM con diferentes valores por entorno.

#### **Ventajas**

- No depende del sistema operativo.
- Se puede usar el formato (yaml, json, etc.) que más se adecue al desarrollo.
- El mismo archivo de configuración puede ser compartido por diferentes nodos.
- Soporte nativo por parte de los lenguajes de programación para el consumo de los archivos.

#### **Desventajas**

- Para poder utilizar el mismo archivo a través de diferentes nodos, tiene que haber interconectividad entre estos a través de herramientas para compartir directorios.

- No cuenta con un sistema de versionamiento.
- Falta de mecanismos de seguridad.

### **2.2.1.3. Sistemas de configuración**

La tercera alternativa es a través del uso de servidores especiales (en la nube u on-premise). Dependiendo del sistema de configuración, este puede almacenar la configuración en una estructura de clave/valor.

Los sistemas de configuración poseen características avanzadas como el rastreo de cambios, servicios de respaldo y validaciones para la detección de fallas.

#### **Ventajas**

- Actualización de valores en tiempo de ejecución.
- Control de versiones de las claves y valores.
- Acceso centralizado a través del protocolo http.
- Soporte para agregar mecanismos de seguridad.

#### **Desventajas**

- Requiere de clases utilitarias para consumir los valores almacenados.

### 2.3.2. Sistemas de software existentes.

Actualmente, existen una gran variedad de herramientas que implementan la externalización de la configuración, y debido a ello, los arquitectos de software tienen un difícil trabajo para seleccionar la herramienta que mejor se adecue a su entorno y que cumpla con sus requerimientos específicos.

A continuación, se listan algunas herramientas existentes, las cuales fueron obtenidas como resultado de la investigación.

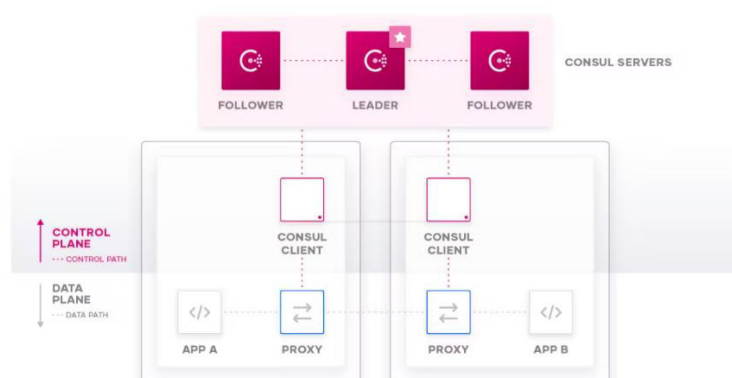
#### 2.3.2.1. Consul

Es una herramienta que permite el “descubrimiento de servicios”, “comprobación de salud” y el “almacenamiento de la configuración usando una estructura de clave/valor” (*What is Consul? | Consul | HashiCorp Developer, 2023*).

Está diseñado para ser amigable con la comunidad DevOps y los desarrolladores, haciéndolo perfecto para infraestructuras modernas y elásticas (*What is Consul? | Consul | HashiCorp Developer, 2023*), pero para conseguir esto, ha tenido que convertirse en un servidor completo, que cuenta con sus propias reglas de extensibilidad, escalamiento a través de agentes como se observa en la Figura 17.

**Figura 17**

*Arquitectura de Consul*



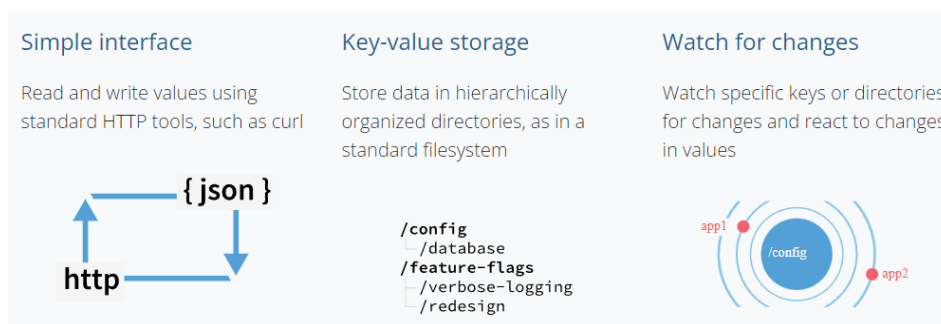
*Nota.* Tomado de *What is Consul?*, por Hashicorp Developer, 2023, Hashicorp (<https://developer.hashicorp.com/consul/docs/intro>).

### 2.3.2.2. Etcd

Es una herramienta que provee un almacenamiento del tipo clave/valor, las principales características se muestran a continuación en la Figura 18.

**Figura 18**

#### *Características de Etcd*



*Nota.* Tomado de *etcd a distributed, reliable key-value store for the most critical data of a distributed system*, por Etcd, 2023, etcd (<https://etcd.io/>).

- Exposición de una API REST para gestionar los valores almacenados.
- Almacenamiento con estructura jerárquica.
- Oyente para vigilar los cambios.

### 2.3.2.3. Archaius

Es el framework de configuración utilizado por todos los componentes de Netflix OSS, el cual es una extensión del proyecto “Apache Common Configuration”.

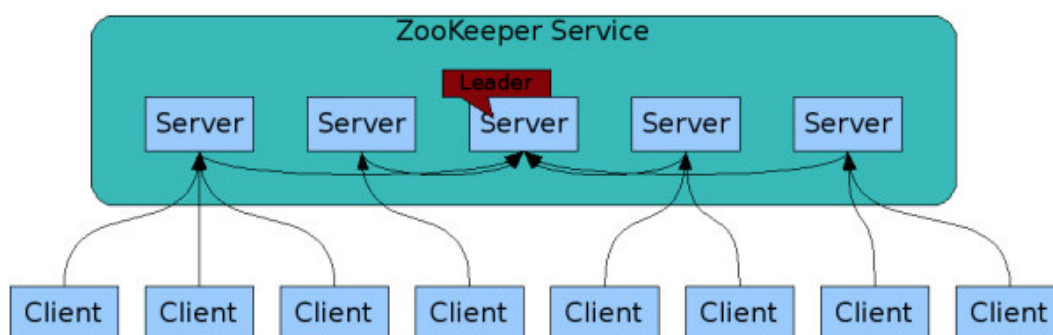
Archaius se basa en el concepto de configuración compuesta, es decir que cada configuración puede ser obtenida de diferentes fuentes a la vez, como: JDBC, REST, archivos properties, etc. Adicionalmente permite la consulta de la configuración en tiempo de ejecución. (*Netflix/archaius*, 2012/2023)

### 2.3.2.4. Zookeeper

Es un servicio centralizado con el propósito de mantener, nombrar y proporcionar sincronización distribuida para la información de la configuración. (*ZooKeeper: Because Coordinating Distributed Systems is a Zoo*, 2023)

**Figura 19**

*Zookeeper service*



*Nota.* Tomado de *ZooKeeper: A Distributed Coordination Service for Distributed Applications*, for Zookeeper, 2023, Zookeeper

(<https://zookeeper.apache.org/doc/r3.8.1/zookeeperOver.html>)

Zookeeper fue diseñado como muestra La Figura 19 con el objetivo de cumplir con los siguientes puntos.

- **Simplicidad**, a través de una interfaz de usuario simplificada y centralizada.
- **Replicación**, trabaja como un proceso distribuido donde pueden existir muchos nodos.
- **Ordenado**, las transacciones en Zookeeper están relacionadas a un número secuencial.
- **Rápido**, se ejecuta sobre miles de nodos, lo cual hace posible un consumo de información en menos tiempos al contar siempre con disponibilidad.

### 2.3.2.5. Spring Cloud Config

El framework proporciona soporte al servidor de la configuración y al cliente que consumirá la configuración en un sistema distribuido. El servidor de la configuración provee un lugar central para gestionar las configuraciones externas de las aplicaciones en todos los entornos. (*Spring Cloud Config*, 2023)

Principales características de Spring Cloud Config:

- API HTTP para consumir la configuración.
- Mecanismos de seguridad para almacenar la información.
- Versionamiento de los valores a través del uso de GIT.
- Motor de almacenamiento intercambiable.

### 2.3.2.6. AWS AppConfig

Es un servicio que permite crear, administrar e implementar rápidamente configuraciones de aplicaciones y es parte de la plataforma en la nube de Amazon. (*What Is AWS AppConfig? - AWS AppConfig*, 2023)

El servicio cuenta con las siguientes características:

- **Reducir errores en los cambios de configuración:** Permite crear reglas para validar la configuración.
- **Despliega cambios rápidamente:** Simplifica la administración de aplicaciones a escala mediante la implementación de cambios de configuración desde una ubicación central.
- **Actualizar aplicaciones sin interrupciones:** Despliega los cambios de configuración en tiempo de ejecución, sin un proceso de compilación o dejando fuera de servicio a la aplicación.
- **Controlar el despliegue de cambios en su aplicación:** El servicio permite minimizar el riesgo mediante uso de estrategias de despliegue,

es decir se puede determinar qué tan rápido las aplicaciones obtendrán los nuevos valores de configuración.

### 2.3.2.7. Azure App Configuration

Al igual que la solución de **Amazon**, **Azure App Configuration** proporciona un servicio que centraliza la configuración de la aplicación. (mcleanbyron, 2023)

El servicio ofrece los siguientes beneficios:

- Un servicio completamente administrado que se puede configurar en minutos.
- Representación flexible de claves y mapeos.
- Etiquetado con labels.
- Versionamiento.
- Interfaz de usuario dedicada para la gestión de Feature Flags.
- Seguridad mejorada a través de identidades administradas por Azure
- Cifrado de información sensible.
- Integración nativa con frameworks populares.
- Exposición de API REST.

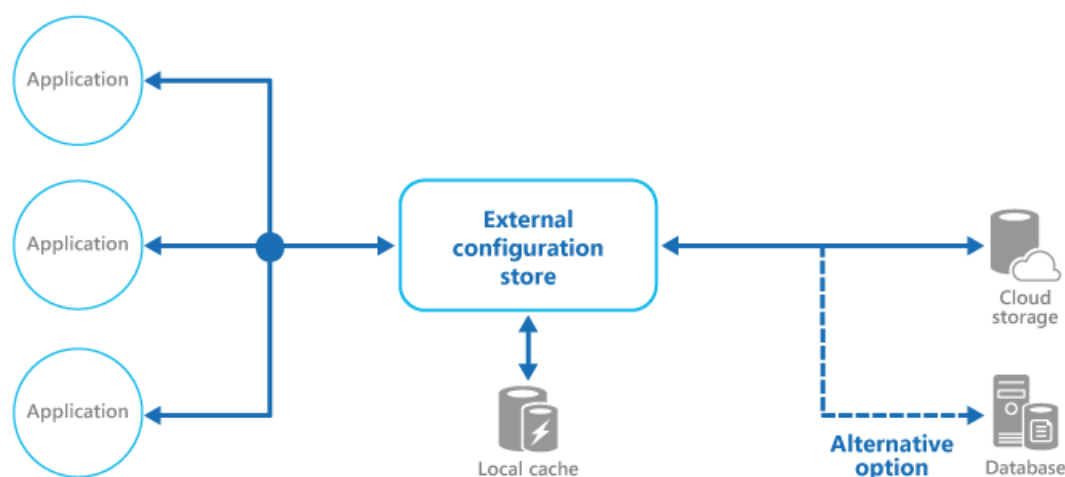
### 2.3.3. Patrón Externalización de la Configuración

Mover la información de configuración fuera del paquete de despliegue de la aplicación a una ubicación centralizada puede proporcionar oportunidades para facilitar la administración y para compartir datos de configuración entre las aplicaciones y las instancias de aplicación. (ckittel, 2023).

En la Figura 20 se observa el diseño de un almacén de la configuración, que se alimenta de dos fuentes externas y a su vez hace uso de cache.

**Figura 20**

*Externalización de la configuración*



Nota. Tomado de *External Configuration Store pattern* – Azzure Architecture Center, por ckittel, 2023, Microsoft (<https://learn.microsoft.com/en-us/azure/architecture/patterns/external-configuration-store>).

Los patrones de diseño dan solución a diversos problemas de desarrollo y este no está exento de ello, ya que su propósito es habilitar un servicio o aplicación para que el mismo artefacto pueda correr en múltiples entornos (debe, test, staging, prod) sin modificación/ recompilación, ya que como es sabido las aplicaciones usan diferentes configuraciones según el entorno donde es desplegado, como la URL de la base de datos, los servicios de terceros, etc. (*Microservices Pattern*, 2023).

### 2.3.4. El Teorema de CAP

Los autores Lomotey & Deters (2013) describen el teorema de CAP con el objetivo de obtener un consumo confiable de servicios web.

Se espera que los servicios web que se ejecutan en sistemas escalables proporcionen las siguientes características:

**Consistencia**, la característica que garantiza que los estados almacenados en un sistema distribuido sean vistos de igual forma en cada nodo por los clientes.

Para asegurar un sistema consistente, los investigadores optan por la técnica ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad).

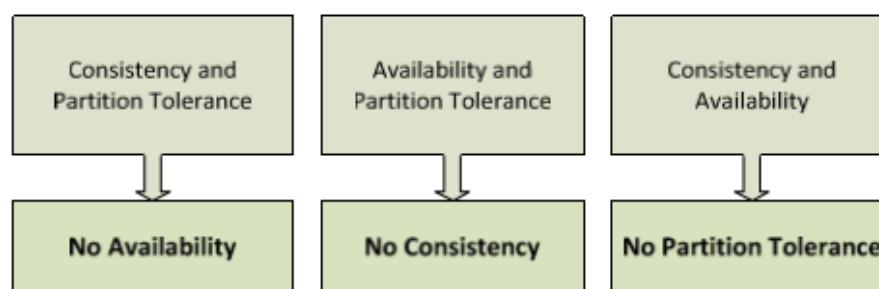
**Disponibilidad**, asegura que cuando una de las partes de los nodos en un sistema distribuido sea inaccesible como el resultado de una falla, los otros nodos deberían seguir operando.

**Tolerancia de partición**, permite que los estados del sistema se mantengan en diferentes ubicaciones.

Sin embargo, ningún sistema distribuido es capaz de garantizar las tres características al mismo tiempo, solo dos de los 3 pueden ser garantizados simultáneamente, como se muestra en la Figura 21. (Lomotey & Deters, 2013).

**Figura 21**

*Las posibles opciones según el teorema de CAP*



*Nota.* Tomado de “Reliable Consumption of Web Services in a Mobile-Cloud Ecosystem Using REST”, por Lomotey & Deters, 2013.

### 2.3.5. Arquitectura Multitenant

IBM (2023) describe la arquitectura multitenant como una arquitectura donde una sola instancia de una aplicación de software sirve a varios tenants, donde un tenant puede ser un usuario individual o un grupo de usuarios que actúan como una organización de clientes que comparten acceso y privilegios comunes dentro de la instancia de la aplicación.

Por naturaleza en esta arquitectura, los datos están aislados e invisible para otros tenants que comparten la instancia de la aplicación con el objetivo de garantizar la seguridad y privacidad de los datos de todos los tenants. (*What Is Multi-Tenant* | IBM, 2023).

A diferencia de las arquitecturas donde se aprovisiona una propia instancia para cada tenant, la arquitectura multitenant ofrece los siguientes beneficios significativos a los proveedores de software.

**Costos más bajos**, debido a que el proveedor de software puede atender múltiples tenants desde una sola instancia de aplicación e infraestructura.

**Escalabilidad**, los tenants pueden escalar a demanda, ya que los nuevos usuarios obtienen acceso a la misma instancia de la aplicación.

**Personalización sin codificación**, Los SaaS son altamente configurables para que cada tenant pueda adaptar la aplicación a su propósito específico.

**Actualizaciones y mantenimiento continuo y constante**, las actualizaciones y parches se aplican a la aplicación sin ningún esfuerzo por parte del cliente y solo una vez.

## CAPITULO 3 METODOLOGIA

### 3.1. Tipo y diseño de la investigación

El presente trabajo de Tesis es una investigación **Aplicada** del tipo **No Experimental**, debido a la naturaleza del proyecto que consiste en implementar el framework “config-manager” para la externalización de la configuración en entornos multitenant, en el cual no cambiarán las variables.

Dado que la investigación No Experimental es realizada sobre situaciones ya existentes, donde las variables no pueden ser manipuladas a diferencia de una investigación Experimental donde la situación es preparada deliberadamente. (Hernández Sampieri & Fernández-Collado, 2014).

La investigación adopta un enfoque Cuantitativo, ya que tiene como objetivo medir los atributos de calidad del framework, los cuales se detallan en la tabla de Operacionalización de las variables (Anexo 1). Para garantizar la coherencia y alineación entre los problemas, objetivos e hipótesis se presenta la Matriz de Consistencia (Anexo 2), que organiza estos elementos esenciales para asegurando así la integridad y el enfoque correcto de la investigación.

Considerando que el objetivo del trabajo de investigación es medir la relación de las variables de forma aislada, es decir sin recibir la influencia de alguna variable extraña, se determina que el alcance de la investigación sea del tipo *correlacional*.

### 3.2. Unidad de análisis

Desde la perspectiva metodológica, el *framework* “config-manager” se considera la *unidad de análisis*, en la que se enfocará el estudio. Y la información relevante se obtendrá de los *Profesionales de Tecnología de Información*, quienes actuarán como la *unidad de observación*.

### 3.3. Población de estudio

Conformado por los Profesionales de Tecnología de Información relacionados activamente con el desarrollo de software.

- Profesionales con una experiencia mayor a 5 años en el desarrollo de software.
- Conocimientos básicos de servicios REST, Microservicios, Multitenancy y Java.

### 3.4. Tamaño de la muestra

#### **Muestra:**

Para la muestra se requiere 12 profesionales que se encuentran altamente involucrados con el desarrollo de software.

### 3.5. Selección de muestra

Basado en el muestreo No probabilístico, se hace uso de la técnica de muestreo por conveniencia, donde se considera a los profesionales que están dispuestos participar del estudio.

### 3.6. Técnicas de recolección de datos

La recopilación de datos se realiza a través de la **encuesta** post uso PSSUQ (Anexo 3 Encuesta de Satisfacción de usuario Post-Uso).

PSSUQ es un instrumento diseñado con el propósito de evaluar la satisfacción percibida por los usuarios con sus sistemas informáticos. (Lewis, 2002).

El PSSUQ ha sido validado en estudios previos, demostrando tanto su fiabilidad como validez. En cuanto a fiabilidad, el instrumento cuenta con valores superiores a 0.85 (coeficiente alfa) haciéndolo idóneo para su uso en investigación y evaluación. En términos de validez, el PSSUQ también ha obtenido un coeficiente de correlación de Pearson significativa de 0.8 lo que indica que mide de manera confiable lo que pretende evaluar.

### **3.7. Análisis e interpretación de la información**

Con el objetivo de analizar e interpretar los datos recolectados con la encuesta PSSUQ (Anexo 3 Encuesta de Satisfacción de usuario Post-Uso), se calculan los promedios de las subescalas Satisfacción, Facilidad de Uso y Eficiencia utilizando las respuestas de los participantes, luego se realiza un análisis descriptivo para calcular la media y varianza, lo que facilita identificar patrones y la dispersión en las percepciones de los usuarios, para finalmente con toda la información obtenida identificar las áreas de mejora para aumentar la satisfacción del usuario.

También se hace uso del alfa de Cronbach para evaluar la fiabilidad del conjunto de preguntas, asegurando que todas ellas estén relacionadas entre sí para medir consistentemente el aspecto que deseamos evaluar.

### 3.8. Metodología de desarrollo de software

La metodología de desarrollo a considerar es la ágil debido a que ha desplazado a las metodologías tradicionales, pero como menciona Matharu et al. (2015) en su trabajo de investigación contamos con un abanico de opciones que difieren en ciertas características y para determinar la metodología ágil que se adecua mejor al proyecto, se debe de realizar un análisis de las características de la Tabla 5.

**Tabla 5**

*Características compatibles de las metodologías ágiles*

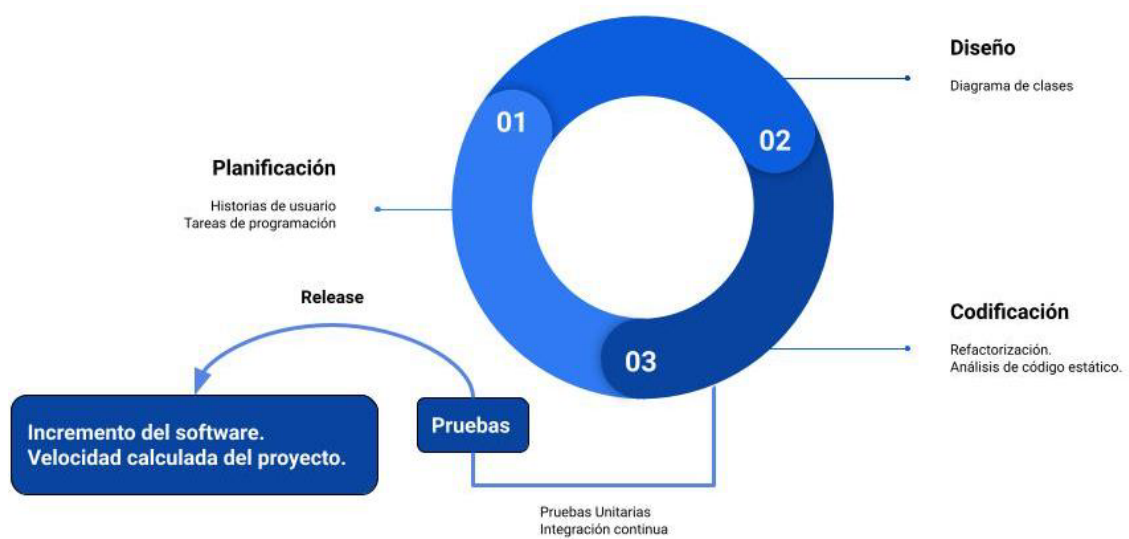
| Parámetros                                   | Scrum                                     | XP                                                                                      | Kanban                                                        |
|----------------------------------------------|-------------------------------------------|-----------------------------------------------------------------------------------------|---------------------------------------------------------------|
| Principios de diseño                         |                                           | Simplificación del código y adaptación de cambios no esperados a través de refactoring. |                                                               |
| Naturaleza de la interacción con el cliente. |                                           | NA                                                                                      |                                                               |
| Complejidad del diseño                       |                                           | Diseño simple                                                                           |                                                               |
| Coordinador del proyecto.                    |                                           | NA                                                                                      |                                                               |
| Roles asignados.                             |                                           | NA                                                                                      |                                                               |
| Dueño del proceso.                           |                                           | NA                                                                                      |                                                               |
| Colaboración del producto.                   |                                           | NA                                                                                      |                                                               |
| Enfoque del flujo de trabajo                 | Iteraciones (sprints)                     | Sin iteraciones. Desarrollo del flujo de trabajo.                                       | Iteraciones cortas.                                           |
| Gestión de requerimientos.                   |                                           | NA                                                                                      |                                                               |
| Entrega del producto                         | Entrega según la duración de los sprints. | Entrega continua.                                                                       | Entrega continua.                                             |
| Estándar de codificación.                    |                                           | Estándares de codificación son usados.                                                  |                                                               |
| Enfoque de pruebas                           |                                           | Desarrollo guiado por pruebas (TDD) y pruebas de aceptación.                            | Pruebas hechas después de la implementación de cada producto. |
| Adaptación a cambios.                        |                                           | Sujeto a cambiar incluso en etapas posteriores del desarrollo.                          |                                                               |

*Nota.* Tomado de “Empirical Study of Agile Software Development Methodologies: A Comparative Analysis”, por Matharu et al., 2015.

A partir de la tabla anterior se puede observar que la metodología XP tiene mayor compatibilidad con el trabajo de investigación, debido a que se hace mayor énfasis en las técnicas que ayudan a reducir la complejidad del desarrollo del software. En base a ello se elaboró la Figura 22 para tener una visión general de las fases de la metodología en el proyecto.

**Figura 22**

*Metodología XP*



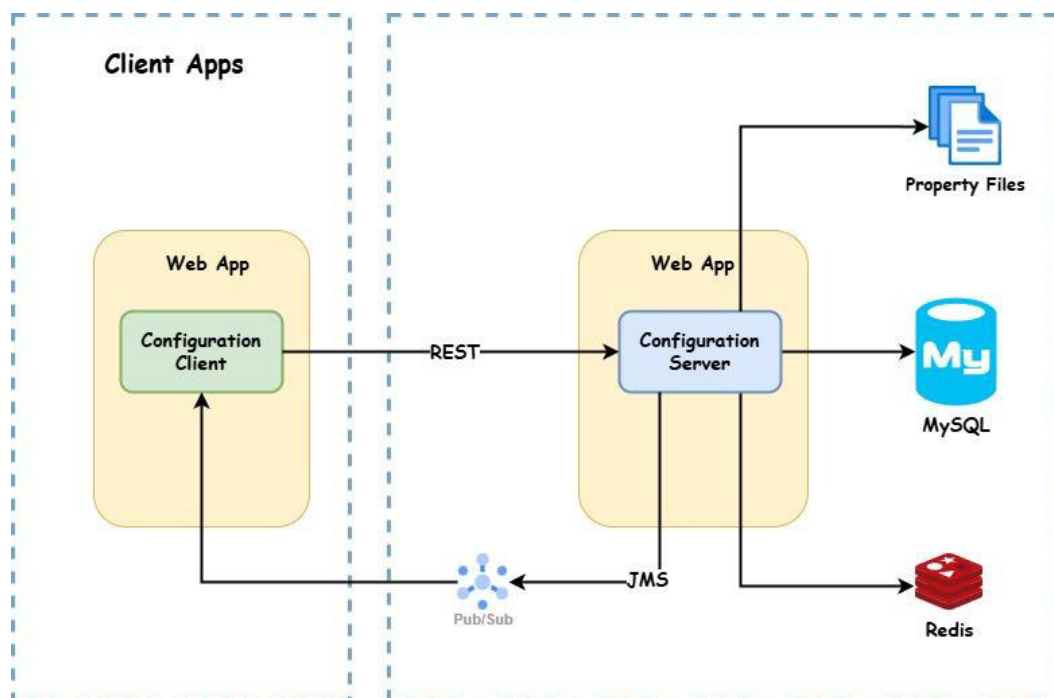
### 3.9. Propuesta Solución

Se propone la creación del framework “config-manager” para la externalización de la configuración que permita a las aplicaciones clientes consumir la configuración de sus proyectos de forma transparente a través de una API REST y actualizar los mismos en tiempo real mediante suscripciones a los Topics (Pub/Sub).

En la Figura 23 se observan las tres fuentes de datos que se puede usar al configurar el framework.

**Figura 23**

*Propuesta solución*



*Nota.* Los proyectos cliente solo pueden configurar uno de los tres tipos de fuente de datos a la vez.

### 3.10. Comparación de herramientas existentes

Actualmente, las diversas soluciones listadas en el marco teórico son utilizadas por las organizaciones en entornos distribuidos para realizar la externalización de la configuración.

Estas soluciones pueden ser agrupadas por la naturaleza de la implementación en los ambientes de trabajo: Herramientas y Frameworks.

Los Frameworks por su naturaleza de librería de desarrollo tienen que ser incorporados a una nueva aplicación o una existente para poder brindar las características de un servidor de externalización de la configuración, a diferencia de las herramientas que son instaladas y configuradas para trabajar correctamente.

Para el análisis comparativo de las herramientas, se toman en consideración las siguientes características:

1. **Almacenamiento de la Configuración:** Este criterio indica si la herramienta o framework utiliza un motor de almacenamiento propio o es integrable con uno de terceros; así como la capacidad de utilizar diversos motores de almacenamientos.
2. **Gestión de la Configuración:** Indica si la herramienta o framework brinda las siguientes capacidades de gestión:
  - Actualización de valores en tiempo de ejecución. - cuando los valores de la fuente son actualizados, automáticamente los valores que ya poseen los clientes también son actualizados, es decir los valores del servidor de configuración y clientes se encuentran sincronizados.
  - Control de versiones. - almacena el historial de cambios por configuración

- Soporte multitenant. - permite tener diversos valores por configuración, permite agruparlos por tenant y manejar plantillas para la fácil adición.
3. **Cientes:** lista los diversos medios de comunicación entre el cliente y la herramienta para gestionar los valores almacenados.
  4. **Licencia:** Indica si la herramienta o framework es pagado, libre o tiene una edición community.

De acuerdo con la información recolectada, en la Tabla 6 se observa que muy pocas soluciones brindan la posibilidad de integrar la herramienta con diversos motores de almacenamiento, debido a que la mayoría impone una forma nativa de almacenamiento, dejando de lado la reutilización de motores de terceros (actualmente consolidados y robustos). Afectando primordialmente a las organizaciones que ya cuentan con motores de almacenamiento en sus ambientes, pues no podrían reutilizar estas herramientas para almacenar también la configuración.

Por otro lado, la actualización en tiempo real de los valores configurados en los clientes, solo son proveídos por los frameworks, la cual es una característica fundamental para este tipo de herramientas.

Adicionalmente el control de versiones no es implementado por muchas soluciones, el cual expone a grandes problemas de inconsistencia a las soluciones.

También al estar orientado a arquitectura de microservicios, utiliza el protocolo http para la comunicación entre los mismos.

Adicionalmente en la Tabla 6 se asignaron los valores 1 y 0 para indicar si la herramienta cuenta o no con la característica respectivamente, y se realizó la suma total de las características por herramienta, donde se observa que el framework propuesto obtuvo el valor de 7, siendo mayor a las demás herramientas evaluadas.

**Tabla 6***Análisis comparativo de herramientas*

| Características                           | Herramientas |           |          |          |                     |               |                         |                     |
|-------------------------------------------|--------------|-----------|----------|----------|---------------------|---------------|-------------------------|---------------------|
|                                           | Consul       | Zookeeper | EtcD     | Archaïus | Spring Cloud Config | AWS AppConfig | Azure App Configuration | Framework Propuesto |
| <b>Almacenamiento de la Configuración</b> |              |           |          |          |                     |               |                         |                     |
| Motor de almacenamiento intercambiable    | 0            | 0         | 0        | 1        | 1                   | 0             | 0                       | 1                   |
| <b>Gestión de la Configuración</b>        |              |           |          |          |                     |               |                         |                     |
| Actualización de valores en tiempo real.  | 0            | 0         | 0        | 1        | 1                   | 0             | 0                       | 1                   |
| Control de versiones                      | 1            | 1         | 0        | 0        | 1                   | 1             | 1                       | 0                   |
| Soporte multitenant                       | 1            | 0         | 0        | 0        | 0                   | 1             | 1                       | 1                   |
| <b>Clientes</b>                           |              |           |          |          |                     |               |                         |                     |
| Interfaz web                              | 1            | 1         | 0        | 0        | 0                   | 1             | 1                       | 0                   |
| Interfaz de línea de comandos             | 1            | 0         | 1        | 0        | 0                   | 1             | 1                       | 1                   |
| API REST                                  | 1            | 0         | 1        | 0        | 1                   | 1             | 1                       | 1                   |
| JMX                                       | 0            | 1         | 0        | 1        | 0                   | 0             | 0                       | 1                   |
| <b>Licencia</b>                           |              |           |          |          |                     |               |                         |                     |
| Open source y gratuito                    | 1            | 1         | 1        | 1        | 1                   | 0             | 0                       | 1                   |
| <b>TOTAL</b>                              | <b>6</b>     | <b>4</b>  | <b>3</b> | <b>4</b> | <b>5</b>            | <b>5</b>      | <b>5</b>                | <b>7</b>            |

## CAPITULO 4 IMPLEMENTACION DE LA HERRAMIENTA

A continuación, se detalla el proceso de desarrollo del framework propuesto, usando la metodología ágil de desarrollo XP.

### 4.1. Implementación

A continuación, se describe el desarrollo del proyecto a través de las fases de la metodología XP.

#### 4.1.1. Fase de Exploración

En esta fase definimos las historias de usuario, la guía de desarrollo (tecnologías, buenas prácticas y procesos), así como la arquitectura del sistema.

##### 4.1.1.1. Historias de Usuario

Para un correcto entendimiento de los requerimientos del software a continuación en la Tabla 7 se detallan las historias de usuario.

**Tabla 7**

*Lista historias de usuario*

| Nro. | Historia de Usuario                                                                      |
|------|------------------------------------------------------------------------------------------|
| 1    | Consumir la configuración del proyecto almacenada en archivos planos.                    |
| 2    | Consumir la configuración del proyecto almacenada en base de datos relacional.           |
| 3    | Consumir la configuración del proyecto almacenada en base de datos NoSql.                |
| 4    | Gestionar la configuración del proyecto a través de perfiles de desarrollo y aplicación. |
| 5    | Registrar el historial de cambios de todos los eventos.                                  |
| 6    | Facilitar la configuración y uso del framework para la aplicación cliente.               |
| 7    | Permitir el almacenamiento local de la configuración en la aplicación.                   |
| 8    | Permitir la actualización de la configuración en tiempo real.                            |

|    |                                                                           |
|----|---------------------------------------------------------------------------|
| 9  | Gestionar la configuración del proyecto a través de tenants.              |
| 10 | Permitir la creación de plantillas para dinamizar la creación de tenants. |
| 11 | Facilitar el consumo de APIs y personalización de servicios,              |

A continuación, se detallan las primeras 4 historias de usuario en la Tabla 8, Tabla 9,

Tabla 10 y Tabla 11 respectivamente y las demás pueden visualizarse en el Anexo 4 Historias de Usuario.

## Tabla 8

### *Historia de usuario 1*

| Historia de Usuario                                                                                                          |                                    |
|------------------------------------------------------------------------------------------------------------------------------|------------------------------------|
| <b>Número:</b> 1                                                                                                             | <b>Usuario:</b> Aplicación Cliente |
| <b>Nombre de Historia:</b> Consumir la configuración del proyecto almacenada en archivos planos.                             |                                    |
| <b>Prioridad:</b>                                                                                                            | <b>Riesgo de desarrollo:</b> Alta  |
| <b>Puntos estimados:</b>                                                                                                     | <b>Iteración asignada:</b> 1       |
| <b>Programador responsable:</b> José Mori                                                                                    |                                    |
| <b>Descripción:</b>                                                                                                          |                                    |
| Obtener los datos de configuración almacenados en archivos planos, para su posterior exposición a través de un servicio web. |                                    |
| <b>Criterios de aceptación:</b>                                                                                              |                                    |
| El usuario puede establecer la ruta del directorio donde se encontrarán los archivos a manipular.                            |                                    |
| El usuario puede manipular los datos del archivo a través de una API REST.                                                   |                                    |
| Las operaciones permitidas por el API REST son de lectura, escritura, actualización y eliminación.                           |                                    |
| <b>Observaciones:</b>                                                                                                        |                                    |

Tabla 9

*Historia de usuario 2*

| Historia de Usuario                                                                                                                      |                                    |
|------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------|
| <b>Número:</b> 2                                                                                                                         | <b>Usuario:</b> Aplicación Cliente |
| <b>Nombre de Historia:</b> Consumir la configuración del proyecto almacenada en base de datos relacional.                                |                                    |
| <b>Prioridad:</b>                                                                                                                        | <b>Riesgo de desarrollo:</b> Alta  |
| <b>Puntos estimados:</b>                                                                                                                 | <b>Iteración asignada:</b> 2       |
| <b>Programador responsable:</b> José Mori                                                                                                |                                    |
| <b>Descripción:</b>                                                                                                                      |                                    |
| Obtener los datos de configuración almacenados en una base de datos relaciona, para su posterior exposición a través de un servicio web. |                                    |
| <b>Criterios de aceptación:</b>                                                                                                          |                                    |
| El usuario puede establecer la conexión de la base de datos relacional a manipular.                                                      |                                    |
| El usuario puede manipular los datos de la base de datos relacional a través de una API REST.                                            |                                    |
| Las operaciones permitidas por el API REST son de lectura, escritura, actualización y eliminación.                                       |                                    |
| <b>Observaciones:</b>                                                                                                                    |                                    |
| El driver de conexión no pertenece al framework, es proveído por la aplicación cliente que usa el framework.                             |                                    |

Tabla 10

*Historia de usuario 3*

| Historia de Usuario                                                                                                                  |                                    |
|--------------------------------------------------------------------------------------------------------------------------------------|------------------------------------|
| <b>Número:</b> 3                                                                                                                     | <b>Usuario:</b> Aplicación Cliente |
| <b>Nombre de Historia:</b> Consumir la configuración del proyecto almacenada en base de datos NoSql.                                 |                                    |
| <b>Prioridad:</b>                                                                                                                    | <b>Riesgo de desarrollo:</b> Alta  |
| <b>Puntos estimados:</b>                                                                                                             | <b>Iteración asignada:</b> 3       |
| <b>Programador responsable:</b> José Mori                                                                                            |                                    |
| <b>Descripción:</b>                                                                                                                  |                                    |
| Obtener los datos de configuración almacenados en una base de datos NoSql, para su posterior exposición a través de un servicio web. |                                    |
| <b>Criterios de aceptación</b>                                                                                                       |                                    |
| El usuario puede establecer la conexión de la base de datos NoSql a manipular.                                                       |                                    |
| El usuario puede manipular los datos de la base de datos NoSql a través de una API REST.                                             |                                    |

---

Las operaciones permitidas por el API REST son de lectura, escritura, actualización y eliminación.

**Observaciones:**

El driver de conexión no pertenece al framework, es proveído por la aplicación cliente que usa el framework.

La base de datos NoSql es Redis.

---

## Tabla 11

### *Historia de usuario 4*

| Historia de Usuario                                                                                                                                                          |                                    |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------|
| <b>Número:</b> 4                                                                                                                                                             | <b>Usuario:</b> Aplicación Cliente |
| <b>Nombre de Historia:</b> Gestionar la configuración del proyecto a través de perfiles de desarrollo y aplicación.                                                          |                                    |
| <b>Prioridad:</b>                                                                                                                                                            | <b>Riesgo de desarrollo:</b> Media |
| <b>Puntos estimados:</b>                                                                                                                                                     | <b>Iteración asignada:</b> 4       |
| <b>Programador responsable:</b> José Mori                                                                                                                                    |                                    |
| <b>Descripción:</b>                                                                                                                                                          |                                    |
| Los datos de configuración de las aplicaciones son almacenados de forma jerárquica a través de aplicaciones y perfiles, donde una aplicación puede tener uno o más perfiles. |                                    |
| <b>Criterios de aceptación</b>                                                                                                                                               |                                    |
| La información almacenada es gestionada a través de una estructura de aplicaciones y perfiles.                                                                               |                                    |
| Cada proyecto puede contener uno o más perfiles.                                                                                                                             |                                    |
| El usuario puede manipular los datos por perfil directamente.                                                                                                                |                                    |
| <b>Observaciones:</b>                                                                                                                                                        |                                    |
| La consulta de los valores se realiza con el id de la aplicación y perfil de esta.                                                                                           |                                    |

---

#### 4.1.1.2. Modelo de Vista de Arquitectura

A continuación, se describe la arquitectura del sistema a través de múltiples vistas (4+1)

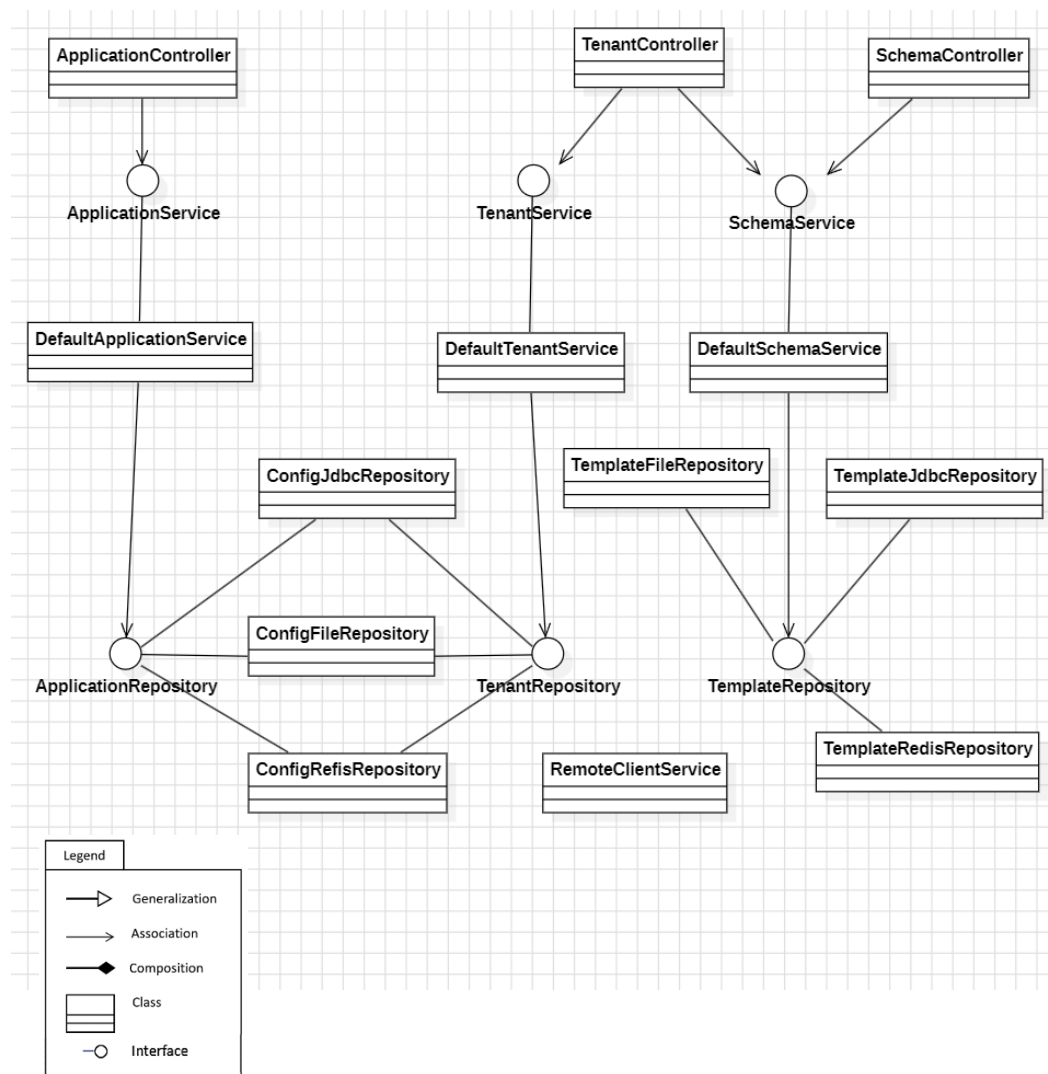
##### 4.1.1.2.1. Vista Lógica.

La vista lógica representada a través del diagrama de clases de forma resumida, sin considerar las operaciones y atributos.

En la Figura 24 se observa el uso de interfaces en las diferentes capas con el objetivo de usar diferentes implementaciones.

**Figura 24**

*Arquitectura propuesta - vista lógica*



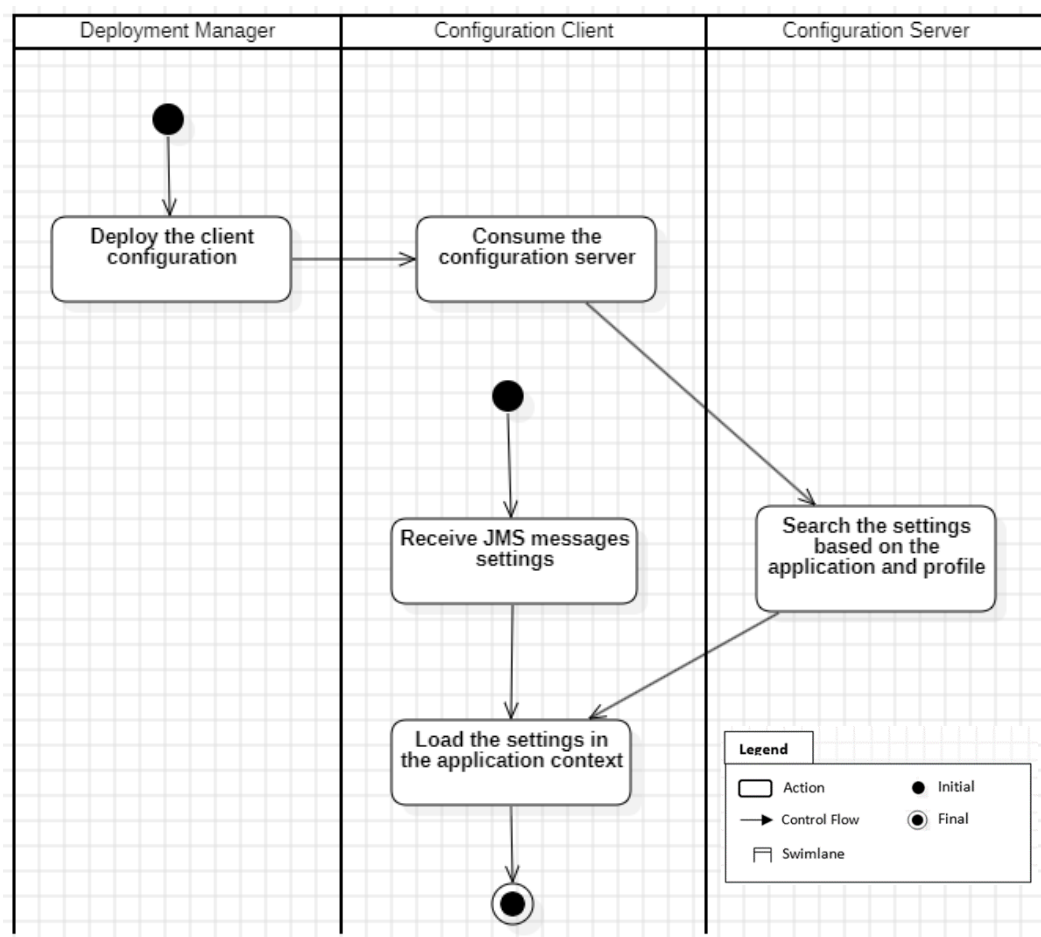
#### 4.1.1.2.2. Vista de Proceso

A través del diagrama de actividades, se puede observar la comunicación de los procesos del sistema.

Desde el punto de vista del cliente, se observan dos puntos de partida, el primero cuando la aplicación es iniciada y el segundo cuando un mensaje es recibido a través de la cola, como se observa en la Figura 25.

**Figura 25**

*Arquitectura propuesta - vista de proceso (client)*

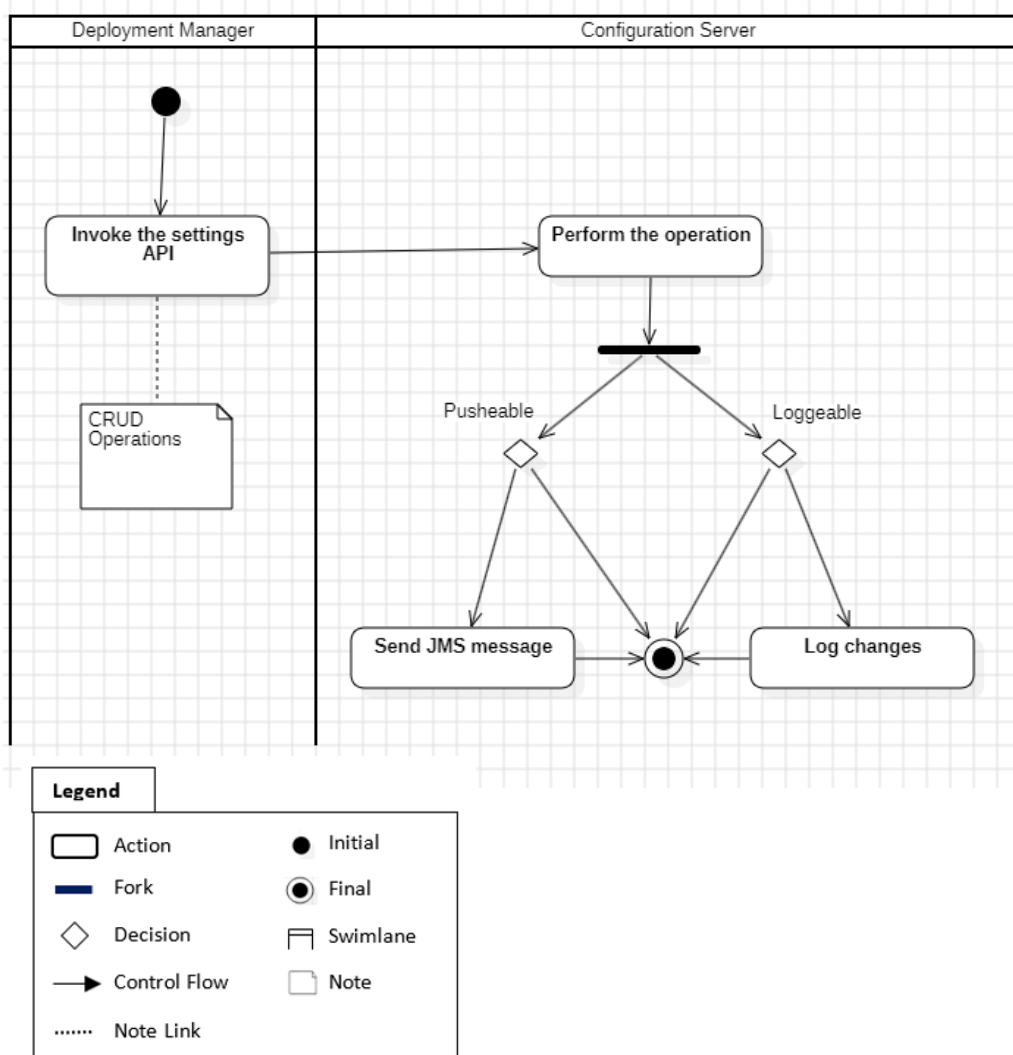


Mientras que, desde el punto de vista del **server**, se observa en la Figura 26 dos actividades condicionadas a la configuración del server,

- Actualización en tiempo real, donde los cambios realizados actualizan los clientes cuando sucede.
- Registro de operaciones, donde todas las acciones realizadas sobre las **configuraciones** son registradas en el log.

**Figura 26**

*Arquitectura propuesta - vista de proceso (server)*



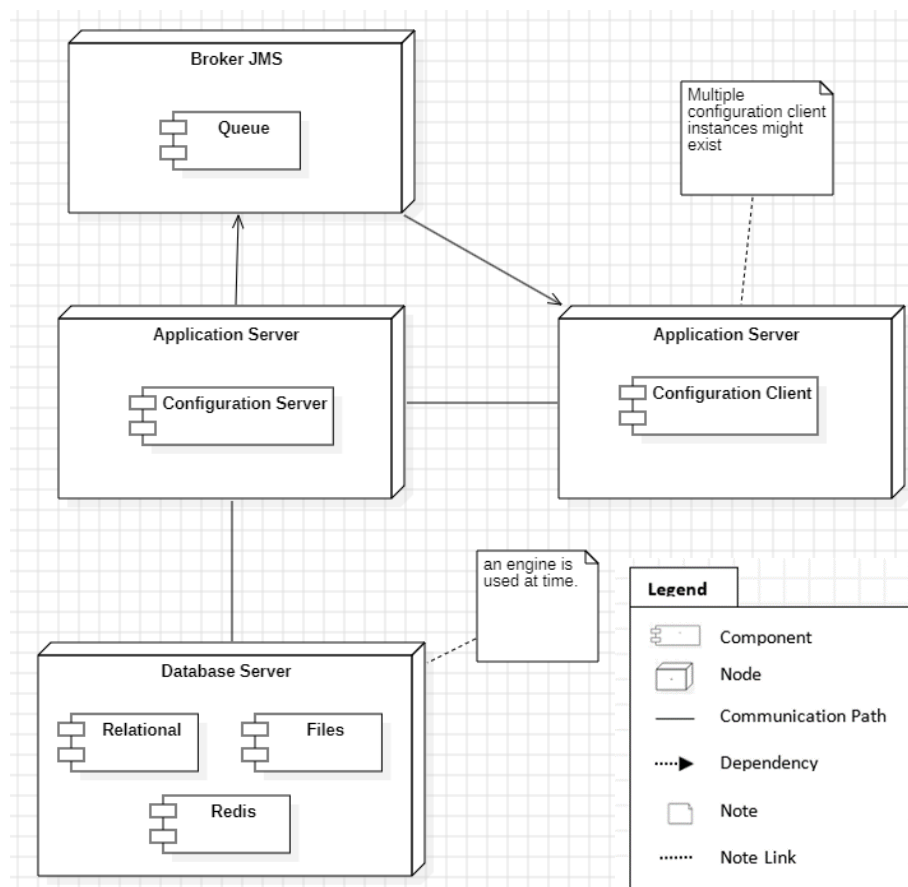
#### 4.1.1.2.3. Vista Física

La vista física representada a través del diagrama de despliegue de la Figura 27 detalla los nodos y componentes involucrados a nivel físico.

- Broker JMS que contiene la cola que es usada para la comunicación entre el **Configuration Server** y **Configuration Client**.
- Application Servers para desplegar el **Configuration Server** y **Configuration Cliente**, donde se considera ambos proyectos pueden tener múltiples instancias de acuerdo con la estrategia de escalamiento del proyecto.
- Database Server, el cual contiene las bases de datos soportadas por la aplicación.

**Figura 27**

*Arquitectura propuesta - vista física*

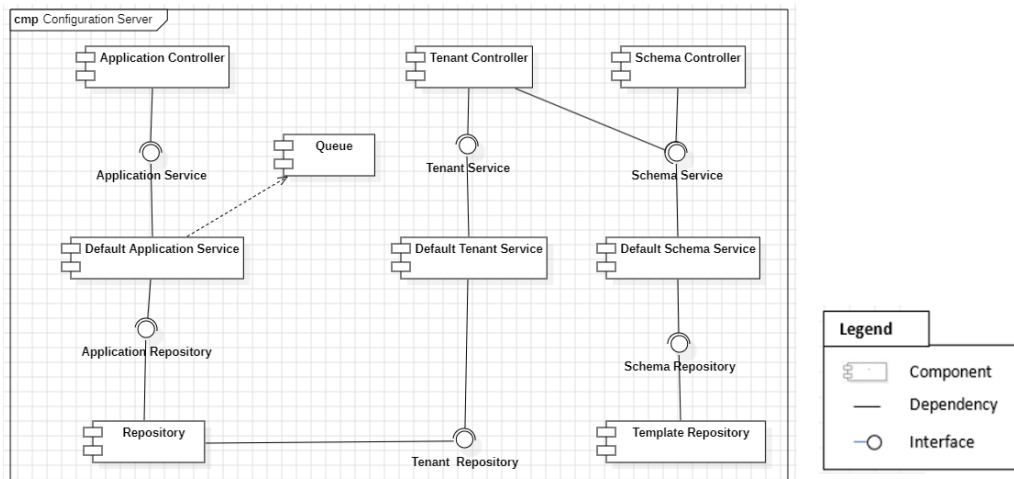


#### 4.1.1.2.4. Vista de Desarrollo

La vista de desarrollo representada por los diagramas de componentes del **Configuration Server** y el **Configuration Client** en la Figura 28 y Figura 29 respectivamente.

**Figura 28**

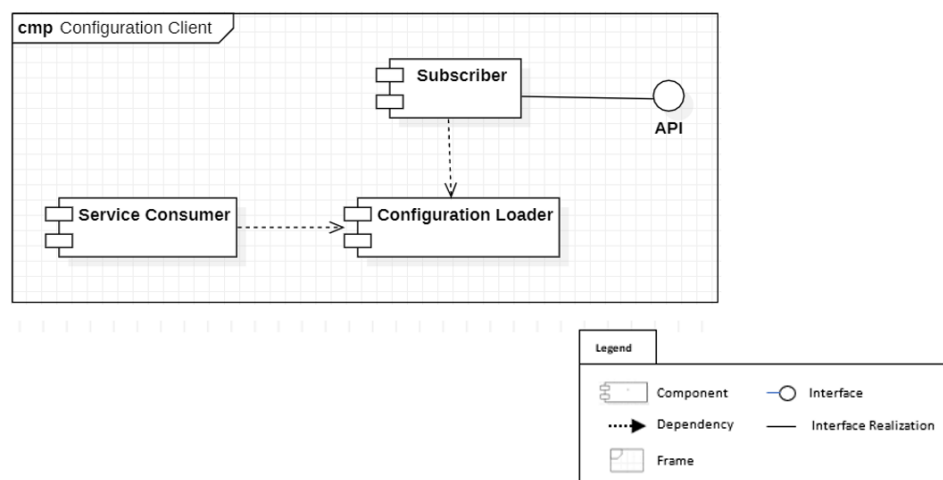
*Arquitectura propuesta - vista de desarrollo (server)*



El componente del **Configuration Client** es simple en comparación al **Configuración Server**, debido a que este solo se encarga de consumir los mensajes proveídos por el segundo.

**Figura 29**

*Arquitectura propuesta - vista de desarrollo (client)*

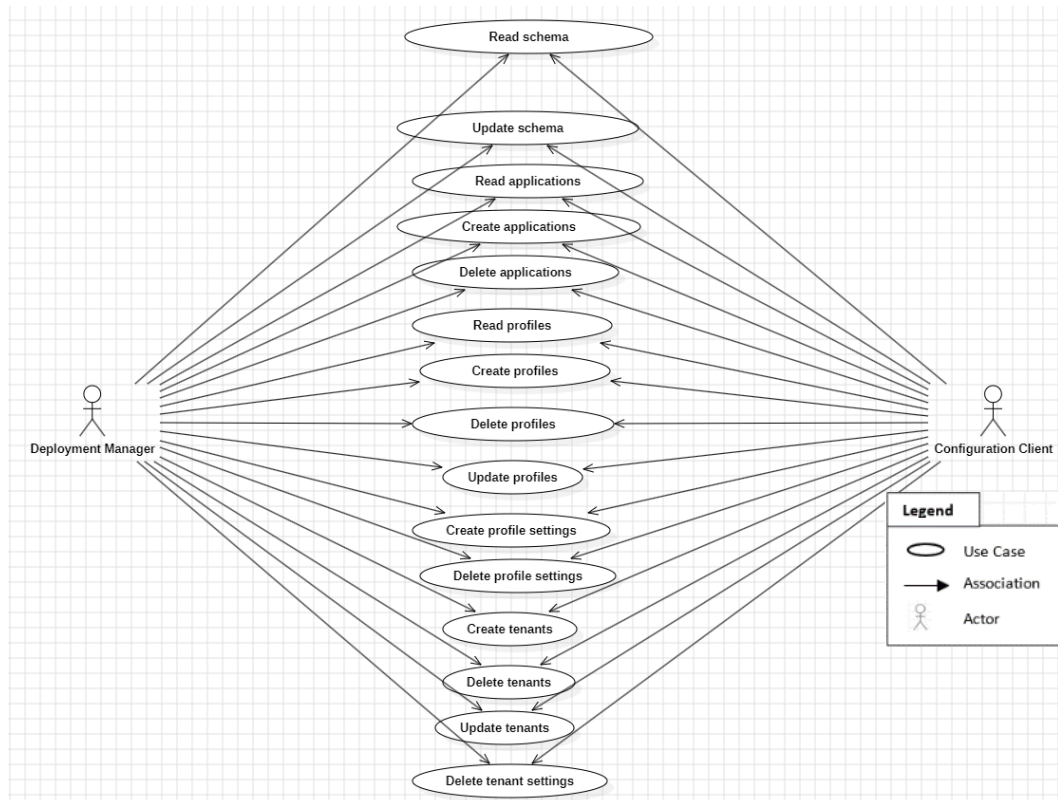


#### 4.1.1.2.5. Escenarios

Se detallan los casos de uso de la aplicación a través del diagrama del mismo nombre en la Figura 30.

**Figura 30**

*Arquitectura propuesta - escenarios*



#### 4.1.1.3. Guía de Desarrollo

Se detallan los procesos y buenas prácticas a considerar para el desarrollo del proyecto.

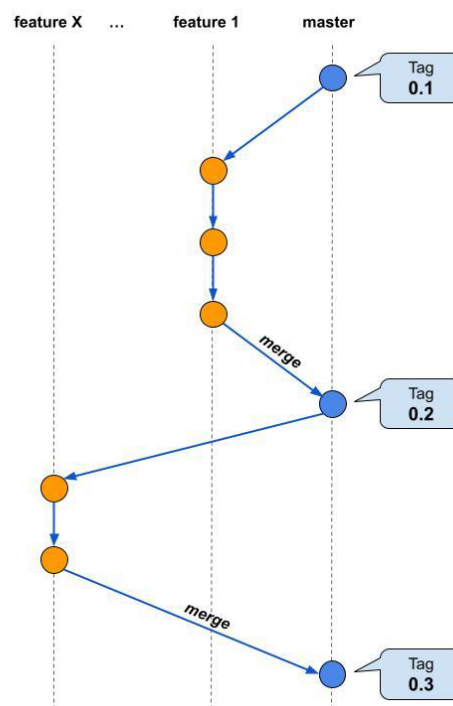
##### 4.1.1.3.1. Versionamiento del código fuente

La gestión del código fuente es una de las principales tareas a tomar en cuenta durante el desarrollo de proyectos, y por ese motivo se tiene que definir un flujo claro de trabajo para tener siempre un código 100% desplegable en la rama principal.

Se ha definido la siguiente estrategia de ramificación en la Figura 31 de acuerdo con la cantidad de desarrolladores y su experiencia.

**Figura 31**

*Estrategia de ramificación*



*Nota.* La figura muestra las ramas utilizadas por la estrategia seleccionada, el Branch master y un Branch por cada característica a implementar.

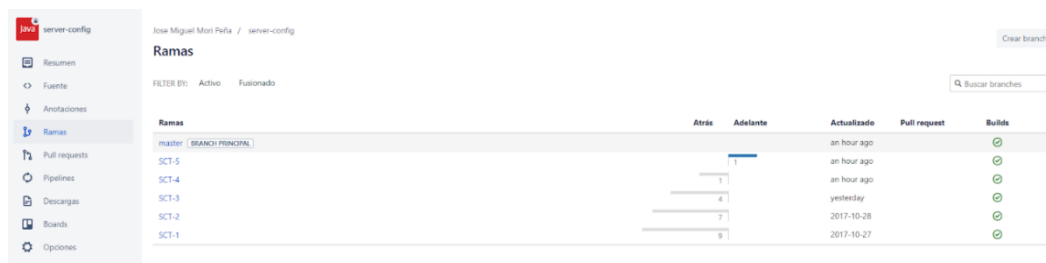
#### 4.1.1.3.2. Flujo de Trabajo

Una rama es creada por tarea de historia de usuario, para realizar la implementación de esta, dicha rama es nombrada usando el identificador de la tarea.

Se pueden observar las ramas de forma visual haciendo uso de la herramienta Bitbucket como se muestra a continuación en la Figura 32.

**Figura 32**

*Bitbucket - Lista de ramas*



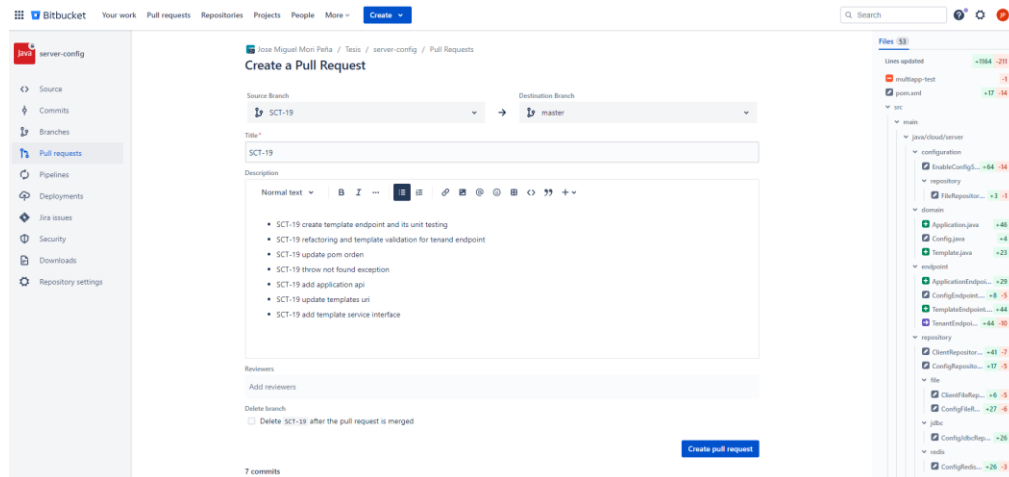
**Nota.** La figura muestra la lista de ramas del proyecto, esta característica la brinda la herramienta bitbucket.

Al culminar el desarrollo de la tarea en la rama, un “Pull Request” es creado, para realizar la fusión del código de la rama de la tarea con la rama principal (master).

La Figura 33 contiene un ejemplo de la creación de un Pull Request en la herramienta Bitbucket.

**Figura 33**

### *Bitbucket - Creación del Pull Request*

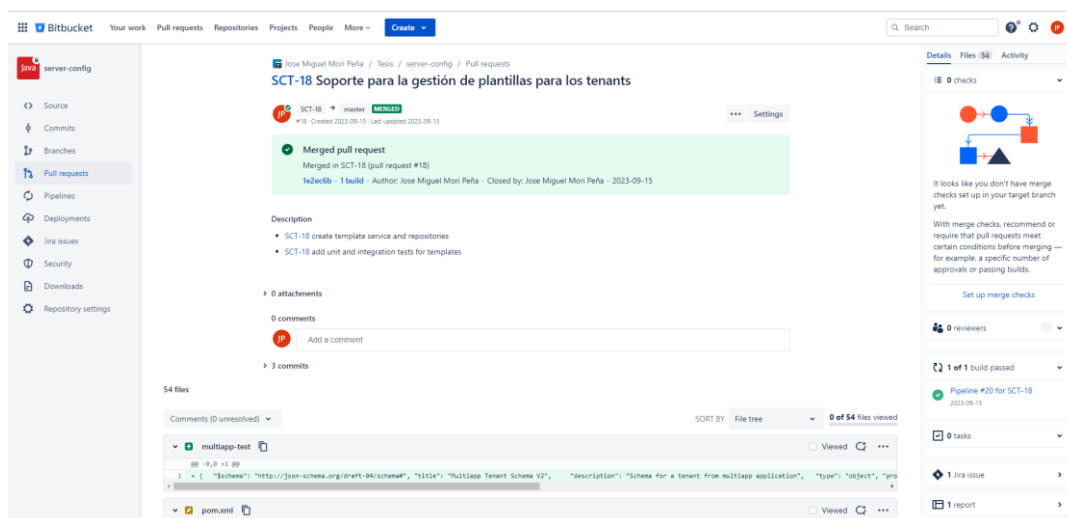


Nota. La figura muestra como la herramienta bitbucket permite la creación de pull requests a través de su UI.

Finalmente se procede a revisar, aprobar y fusionar el nuevo código en la rama principal como se observa en la Figura 34.

**Figura 34**

### *Bitbucket - Revisión de Código*



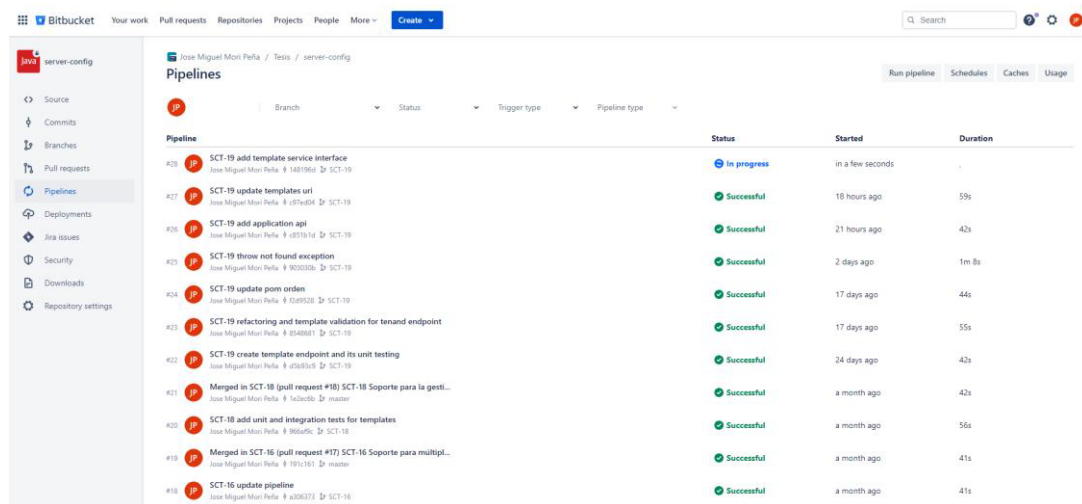
#### 4.1.1.3.3. Integración continua y evaluación de código.

La herramienta Bitbucket ejecutará una tarea de construcción y ejecución de pruebas del proyecto cada vez que se notifique que el código fuente de alguna rama del proyecto ha cambiado, permitiendo saber sobre el estado de salud del proyecto.

La Figura 35 muestra la lista de tareas de construcción ejecutadas de forma automática.

**Figura 35**

*Bitbucket - Integración continua*

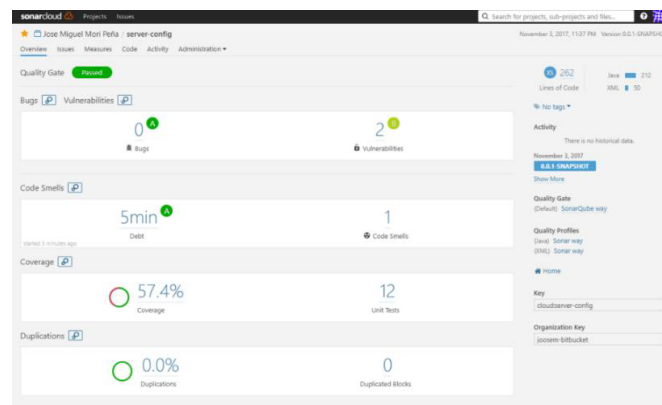


| Pipeline                                                                | Status      | Started          | Duration |
|-------------------------------------------------------------------------|-------------|------------------|----------|
| #20 SCT-19 add template service interface                               | In progress | in a few seconds | -        |
| #27 SCT-19 update templates url                                         | Successful  | 18 hours ago     | 59s      |
| #28 SCT-19 add application api                                          | Successful  | 21 hours ago     | 42s      |
| #29 SCT-19 throw not found exception                                    | Successful  | 2 days ago       | 1m 8s    |
| #34 SCT-19 update pom orden                                             | Successful  | 17 days ago      | 44s      |
| #23 SCT-19 refactoring and template validation for tenant endpoint      | Successful  | 17 days ago      | 55s      |
| #22 SCT-19 create template endpoint and its unit testing                | Successful  | 24 days ago      | 42s      |
| #21 Merged in SCT-18 (pull request #18) SCT-18 Soporte para la gesti... | Successful  | a month ago      | 42s      |
| #20 SCT-18 add unit and integration tests for templates                 | Successful  | a month ago      | 56s      |
| #19 Merged in SCT-16 (pull request #17) SCT-16 Soporte para multipl...  | Successful  | a month ago      | 41s      |
| #18 SCT-16 update pipeline                                              | Successful  | a month ago      | 41s      |

Culminando cada historia de usuario se procede a evaluar el código desarrollado utilizando la herramienta Sonarqube como se visualiza a continuación en la Figura 36.

### Figura 36

## Sonarqube - Evaluación de código



*Nota.* La figura muestra el dashboard de la herramienta Sonarube con los niveles asociados a la evaluación del código en un momento dado.

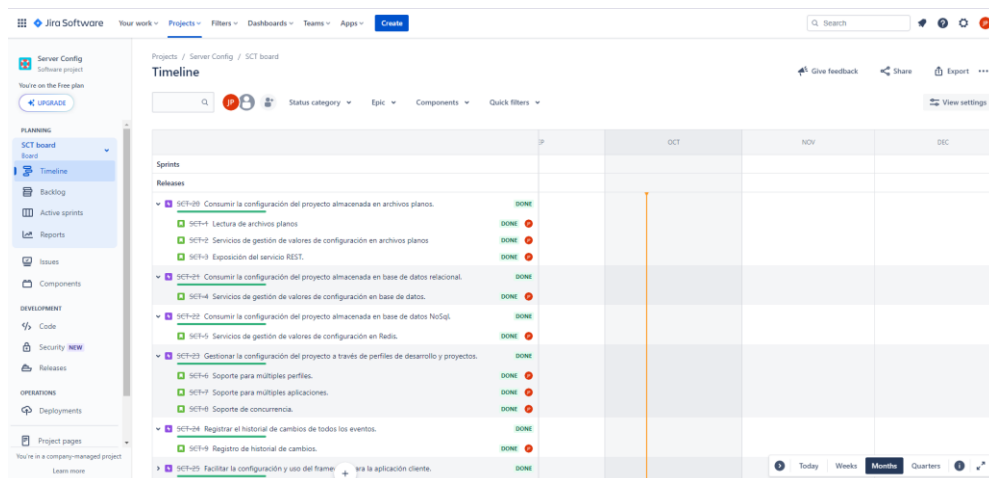
#### 4.1.1.3.4. Seguimiento del proyecto

Las tareas de las historias de usuarios serán gestionadas a través de un tablero Kanban, usando la herramienta Jira.

En la Figura 37 se observan las tareas ya creadas para desarrollar el framework propuesto.

### Figura 37

## Jira - Timeline



### 4.1.2. Fase de Planificación

En esta fase, se establece la prioridad de las historias de usuario y la estimación de estas a través de sus tareas

#### 4.1.2.1. Tareas de Programación

Se define la lista de tareas de programación que permitirá completar las historias de usuarios.

Cada historia de usuario cuenta con una o más tareas de programación. Las primeras 4 tareas de programación se encuentran detalladas en la Tabla 12, Tabla 13, Tabla 14 y Tabla 15 , las demás tareas pueden visualizarse en el Anexo 5 Tareas de Programación

**Tabla 12**

#### *Tarea de programación 1*

| Tarea                                                                                                                    |                     |
|--------------------------------------------------------------------------------------------------------------------------|---------------------|
| Número de tarea: 1                                                                                                       | Número historia: 1  |
| Nombre de tarea: Lectura de archivos planos.                                                                             |                     |
| Tipo de tarea: Desarrollo                                                                                                | Puntos estimados: 5 |
| Fecha inicio:                                                                                                            | Fecha fin:          |
| Programador responsable: José Mori                                                                                       |                     |
| Descripción:                                                                                                             |                     |
| Se leen los valores de los archivos planos de la extensión "properties" en una estructura de datos, para su utilización. |                     |

**Tabla 13***Tarea de programación 2*

| Tarea                                                                                                                   |                            |
|-------------------------------------------------------------------------------------------------------------------------|----------------------------|
| <b>Número de tarea:</b> 2                                                                                               | <b>Número historia:</b> 1  |
| <b>Nombre de tarea:</b> Servicios de gestión de valores de configuración en archivos planos.                            |                            |
| <b>Tipo de tarea:</b> Desarrollo                                                                                        | <b>Puntos estimados:</b> 5 |
| <b>Fecha inicio:</b>                                                                                                    | <b>Fecha fin:</b>          |
| <b>Programador responsable:</b> José Mori                                                                               |                            |
| <b>Descripción:</b>                                                                                                     |                            |
| Se crea el conjunto de servicios (bien estructurados y extensibles) que permita la consulta de los valores almacenados. |                            |

**Tabla 14***Tarea de programación 3*

| Tarea                                                        |                            |
|--------------------------------------------------------------|----------------------------|
| <b>Número de tarea:</b> 3                                    | <b>Número historia:</b> 1  |
| <b>Nombre de tarea:</b> Exposición del servicio REST.        |                            |
| <b>Tipo de tarea:</b> Desarrollo                             | <b>Puntos estimados:</b> 5 |
| <b>Fecha inicio:</b>                                         | <b>Fecha fin:</b>          |
| <b>Programador responsable:</b> José Mori                    |                            |
| <b>Descripción:</b>                                          |                            |
| Se crea una API REST para consultar los valores almacenados. |                            |

Tabla 15

*Tarea de programación 4*

| Tarea                                                                                                                   |                      |
|-------------------------------------------------------------------------------------------------------------------------|----------------------|
| Número de tarea: 4                                                                                                      | Número historia: 2   |
| Nombre de tarea: Servicios de gestión de valores de configuración en base de datos.                                     |                      |
| Tipo de tarea: Desarrollo                                                                                               | Puntos estimados: 21 |
| Fecha inicio:                                                                                                           | Fecha fin:           |
| Programador responsable: José Mori                                                                                      |                      |
| Descripción:                                                                                                            |                      |
| Se crea el conjunto de servicios (bien estructurados y extensibles) que permita la consulta de los valores almacenados. |                      |

#### 4.1.2.2. Priorización de las Historias de Usuario.

En esta fase, se procede a asignarle prioridades a las historias de usuario con el objetivo de conocer el orden en la cual las tareas serán desarrolladas en la fase de iteraciones.

Las historias de usuario que hacen referencia características base son consideradas de alta prioridad con respecto a las otras como se detalla en la Tabla 16.

**Tabla 16**

*Lista priorizada de historias de usuario*

| Nro. | Historia de Usuario                                                                     | Prioridad |
|------|-----------------------------------------------------------------------------------------|-----------|
| 1    | Consumir la configuración del proyecto almacenada en archivos planos.                   | Alta      |
| 2    | Consumir la configuración del proyecto almacenada en base de datos relacional.          | Alta      |
| 3    | Consumir la configuración del proyecto almacenada en base de datos NoSql.               | Alta      |
| 4    | Gestionar la configuración del proyecto a través de perfiles de desarrollo y proyectos. | Alta      |
| 5    | Registrar el historial de cambios de todos los eventos.                                 | Alta      |
| 6    | Facilitar la configuración y uso del framework para la aplicación cliente.              | Media     |
| 7    | Permitir el almacenamiento local de la configuración en la aplicación cliente.          | Media     |
| 8    | Permitir la actualización de la configuración en tiempo real.                           | Media     |
| 9    | Gestionar la configuración del proyecto a través de tenants.                            | Media     |
| 10   | Permitir la creación de plantillas para dinamizar la creación de tenants.               | Media     |
| 11   | Facilitar el consumo de APIs y personalización de servicios                             | Media     |

### 4.1.2.3. Estimación de las Historias de Usuario

La estimación del esfuerzo de cada tarea de usuario se realiza utilizando el juicio de experto, tomando como base la complejidad que estas tareas pueden tener. En la Tabla 17 se observan las historias ya estimadas con los puntos respectivos.

**Tabla 17**

*Lista estimada de historias de usuario*

| Nro. | Historia de Usuario                                                                     | Puntos |
|------|-----------------------------------------------------------------------------------------|--------|
| 1    | Consumir la configuración del proyecto almacenada en archivos planos.                   | 13     |
| 2    | Consumir la configuración del proyecto almacenada en base de datos relacional.          | 21     |
| 3    | Consumir la configuración del proyecto almacenada en base de datos NoSql.               | 21     |
| 4    | Gestionar la configuración del proyecto a través de perfiles de desarrollo y proyectos. | 13     |
| 5    | Registrar el historial de cambios de todos los eventos.                                 | 13     |
| 6    | Facilitar la configuración y uso del framework para la aplicación cliente.              | 10     |
| 7    | Permitir el almacenamiento local de la configuración en la aplicación cliente.          | 8      |
| 8    | Permitir la actualización de la configuración en tiempo real.                           | 21     |
| 9    | Gestionar la configuración del proyecto a través de tenants.                            | 18     |
| 10   | Permitir la creación de plantillas para dinamizar la creación de tenants.               | 13     |
| 11   | Facilitar el consumo de APIs y personalización de servicios                             | 21     |

### 4.1.3. Fase de Iteraciones

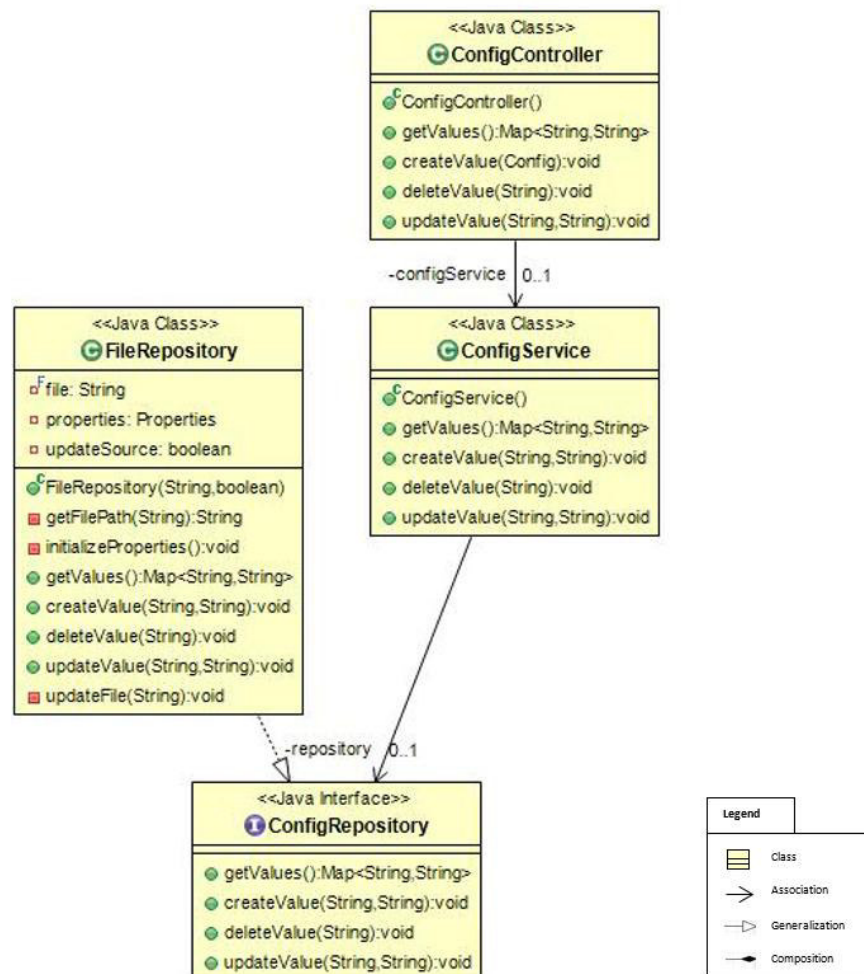
Para cada iteración se realiza el diagrama de clases de las tareas de desarrollo, los cuales van evolucionando hasta obtener a la solución definitiva.

#### 4.1.3.1. Iteración 1

La primera iteración representada en la Figura 38 se centra en el desarrollo de las clases que permiten almacenar y obtener la configuración almacenada en archivos planos. Por otro lado, se desarrolla la primera versión de la API a través de la clase “ConfigController”.

**Figura 38**

*Iteración 1 - Diagrama de clases*

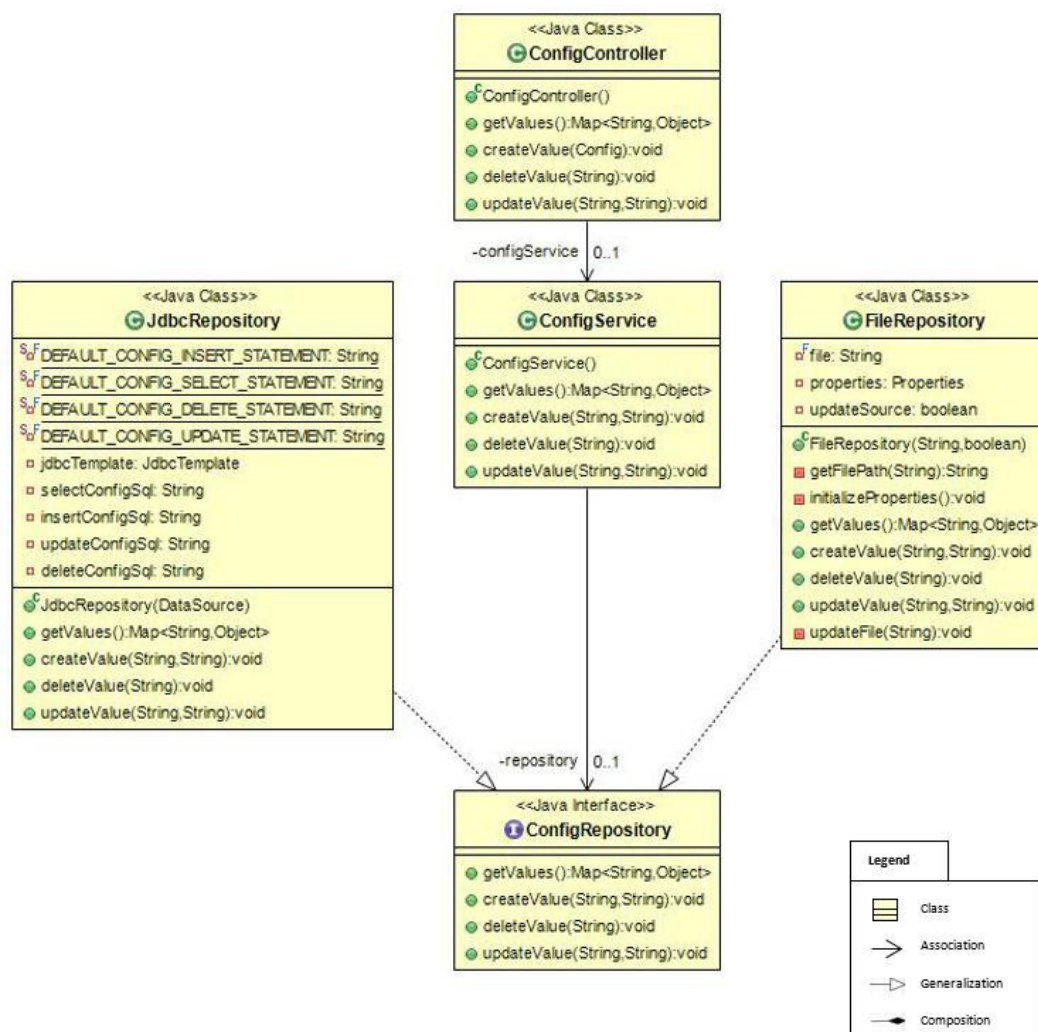


#### 4.1.3.2. Iteración 2

Esta iteración representada en la Figura 39 se centra en el desarrollo de las clases que permiten almacenar y obtener la información de base de datos relacionales.

**Figura 39**

*Iteración 2- Diagrama de clases*

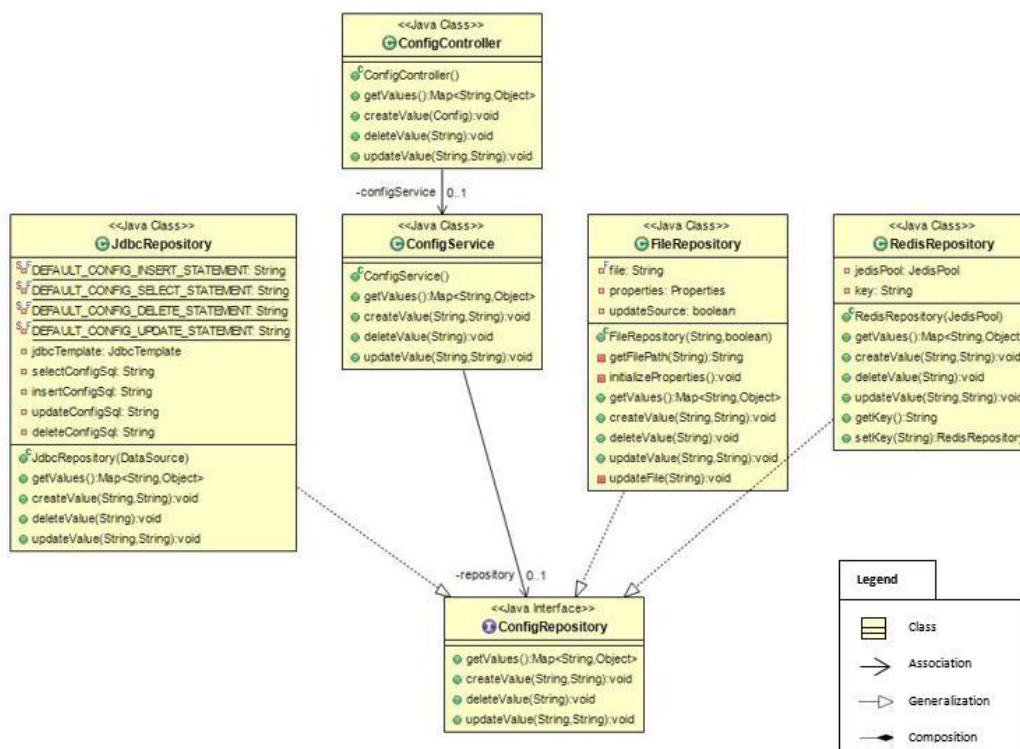


### 4.1.3.3. Iteración 3

Esta iteración se centra en el desarrollo de las clases que permiten almacenar y obtener la información de Redis, como se observa en la Figura 40.

**Figura 40**

*Iteración 3 - Diagrama de clases*

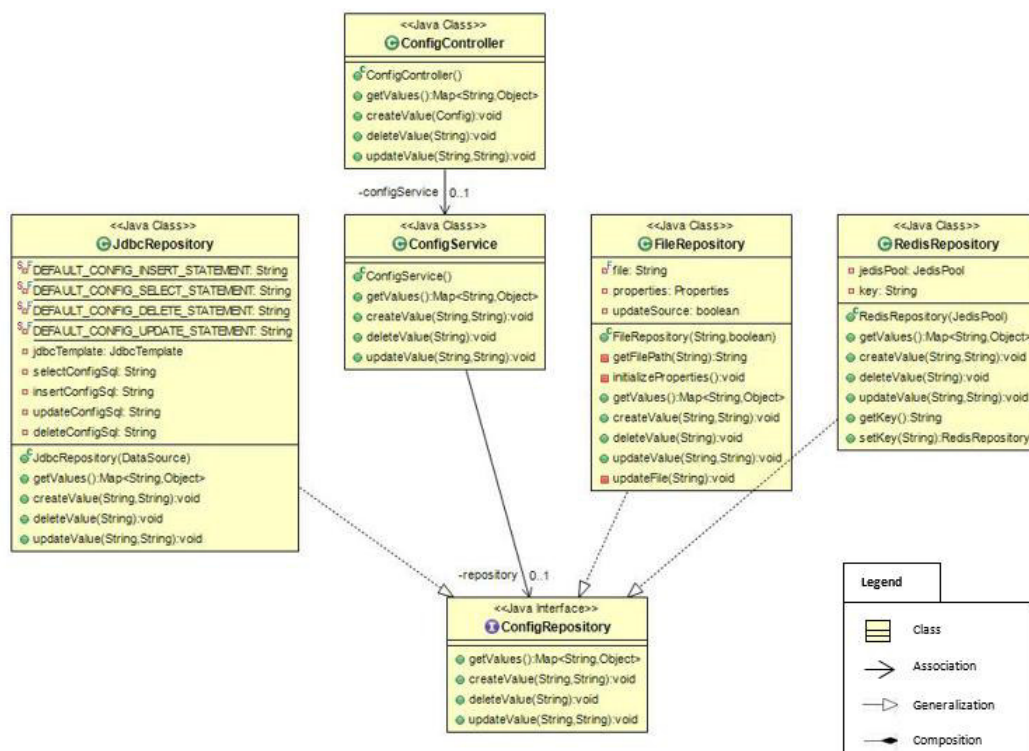


#### 4.1.3.4. Iteración 4

Esta iteración se centra en las modificaciones de las clases actuales para que soporten diversas aplicaciones y perfiles a la vez, como se observa en la Figura 41.

**Figura 41**

*Iteración 4 - Diagrama de clases*

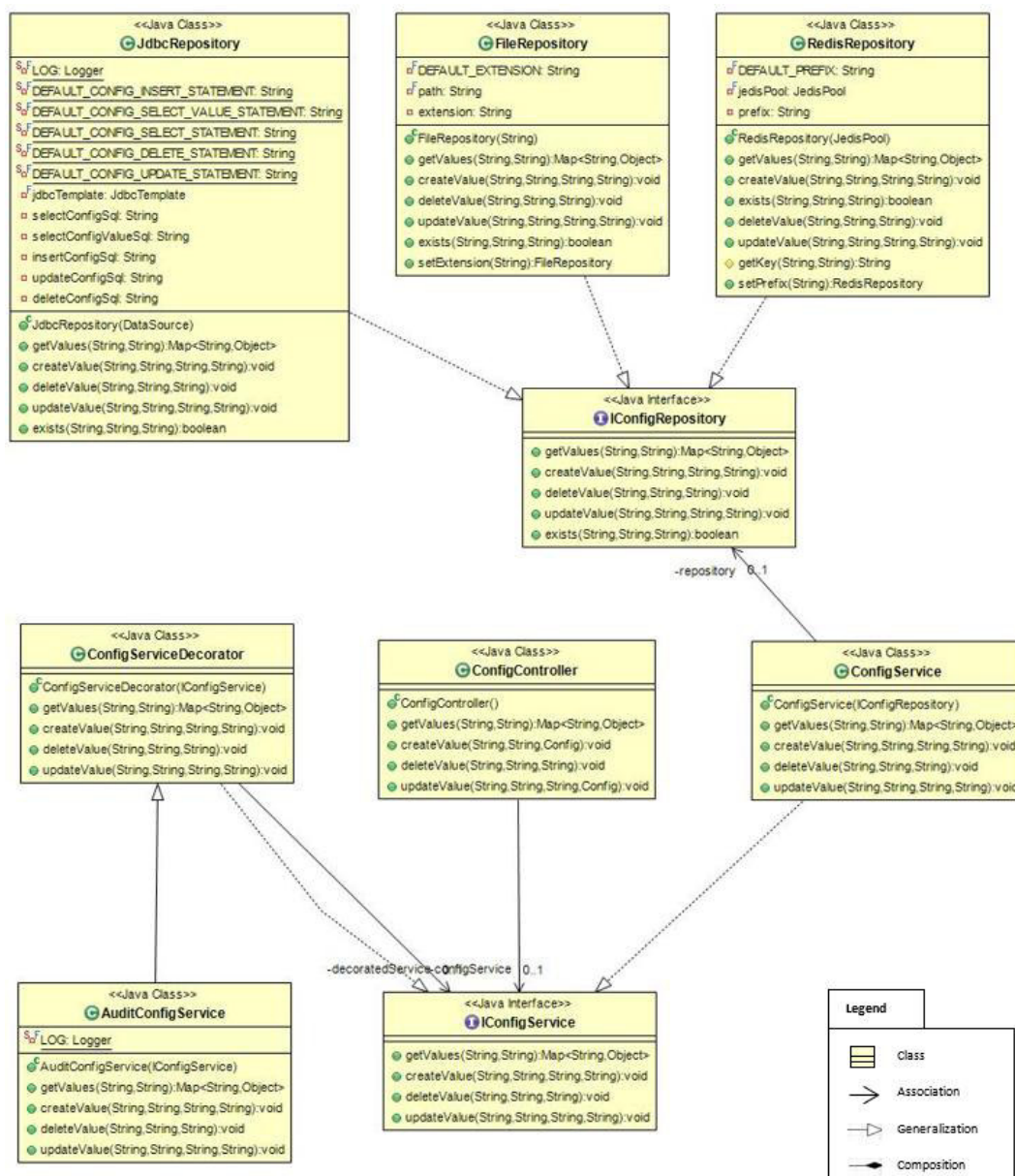


#### 4.1.3.5. Iteración 5

Esta iteración se centra en el desarrollo de las clases que permite registrar el historial de cambios realizados a través de la API, como se observa en la Figura 42.

**Figura 42**

*Iteración 5 - Diagrama de clases*



### 4.1.3.6. Iteración 6

Esta iteración representada en las Figura 43 y Figura 44 se centra en el desarrollo de las clases que permiten configurar el cliente y servidor con el uso de anotaciones.

**Figura 43**

*Iteración 6- Diagrama de clases del “Configuration Server”*

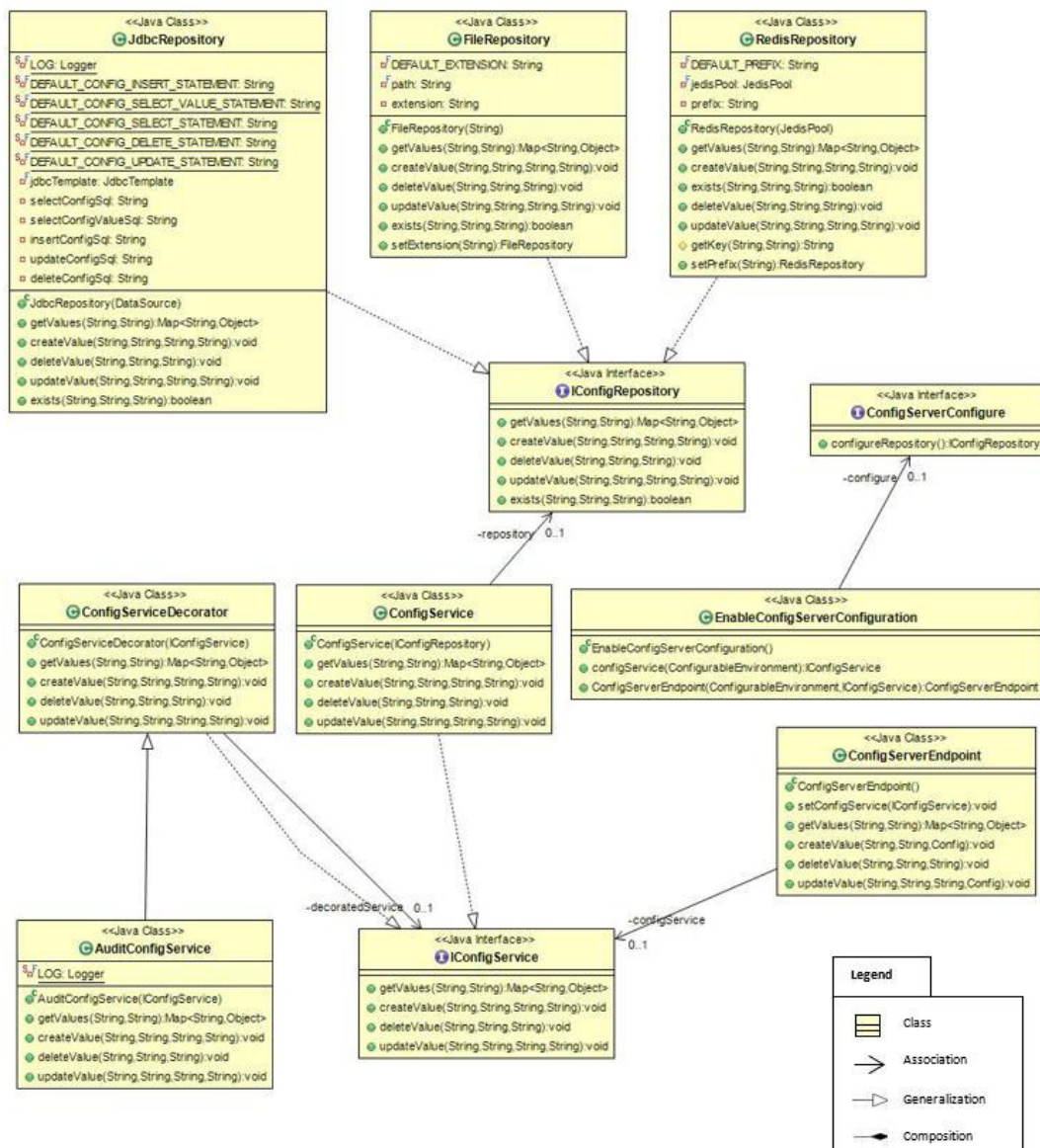
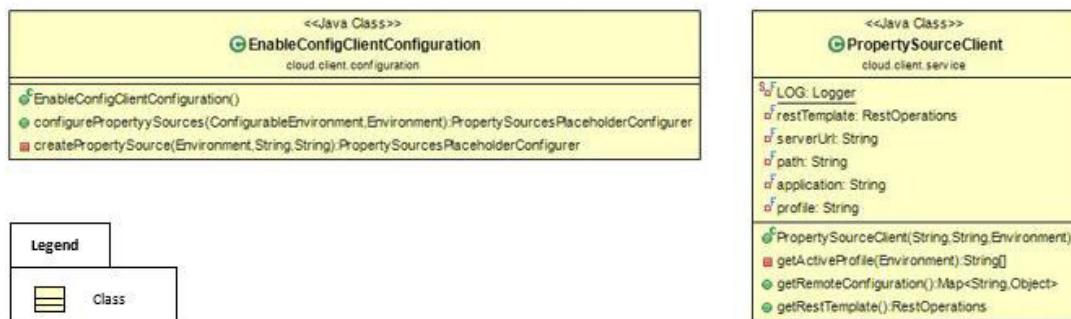


**Figura 44***Iteración 6 - Diagrama de clases de "Configuration Client"*

#### 4.1.3.7. Iteración 7

Esta iteración se centra en el desarrollo de las clases que permiten cargar la información obtenida del servidor, en el contexto de la aplicación cliente, como se observa en la Figura 45.

**Figura 45***Iteración 7 - Diagrama de clases*

#### 4.1.3.8. Iteración 8

Esta iteración se enfoca en darle la capacidad de actualizar los valores de configuración en todos los clientes a través del servidor de la configuración, como se observa en la Figura 46, Figura 47 y Figura 48, los diagramas de clase relacionados al Configuration Server y en la Figura 49 el diagrama de clases relacionado al Configuration Client.

**Figura 46**

*Iteración 8 - Diagrama de clases del "Configuration Server" I*

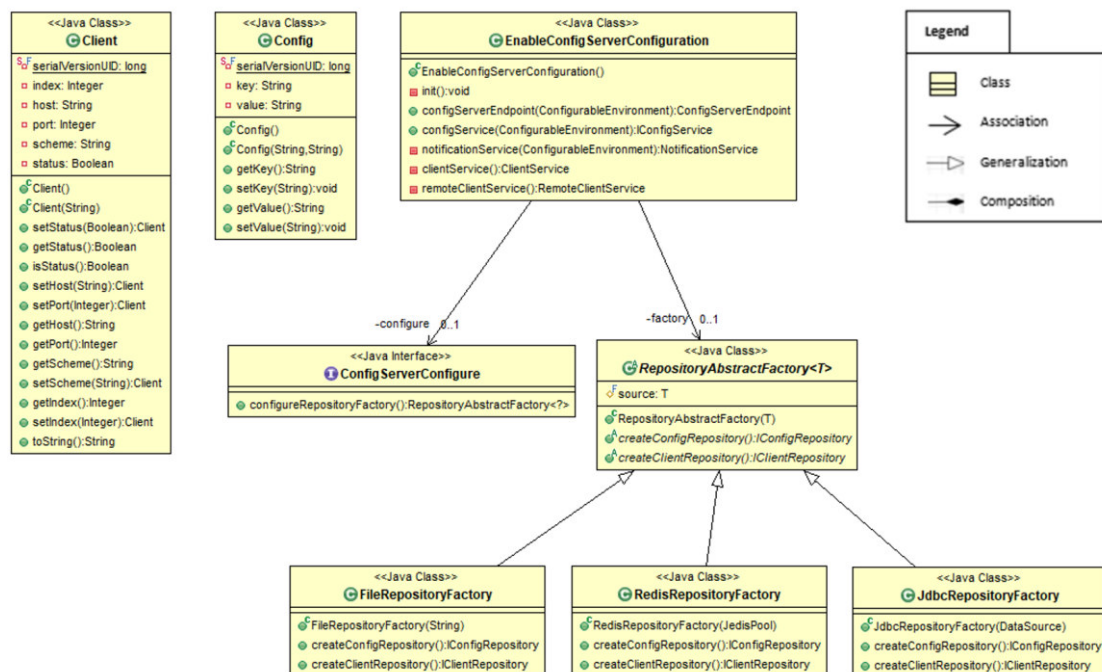


Figura 47

*Iteración 8 - Diagrama de clases del "Configuration Server" II*

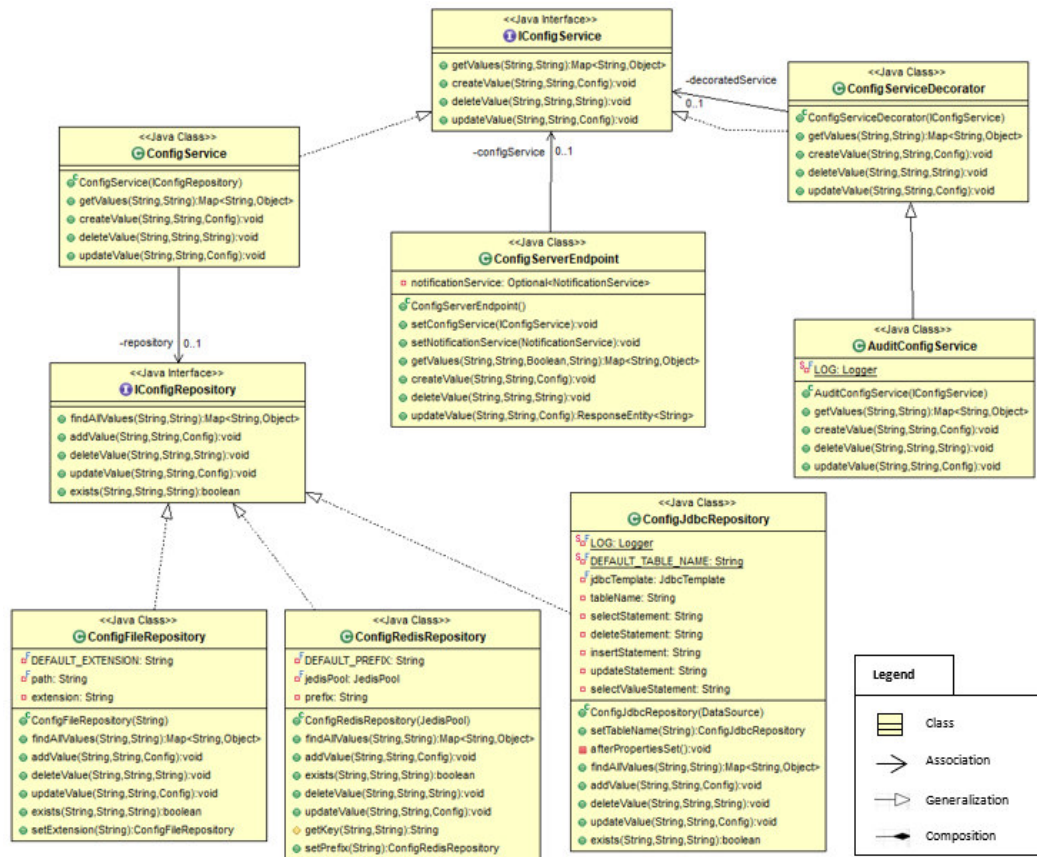


Figura 48

## Iteración 8 - Diagrama de clases del "Configuration Server" III

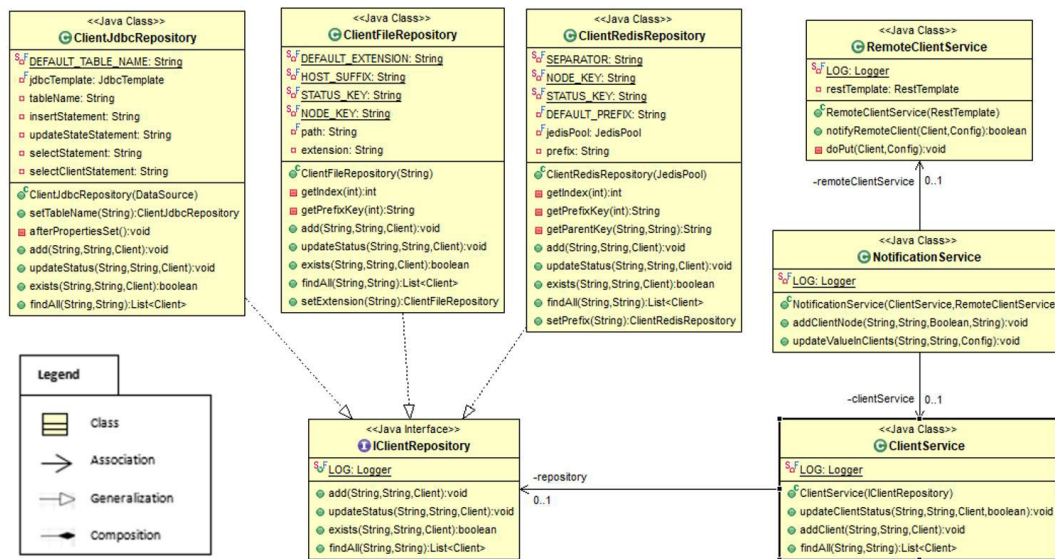
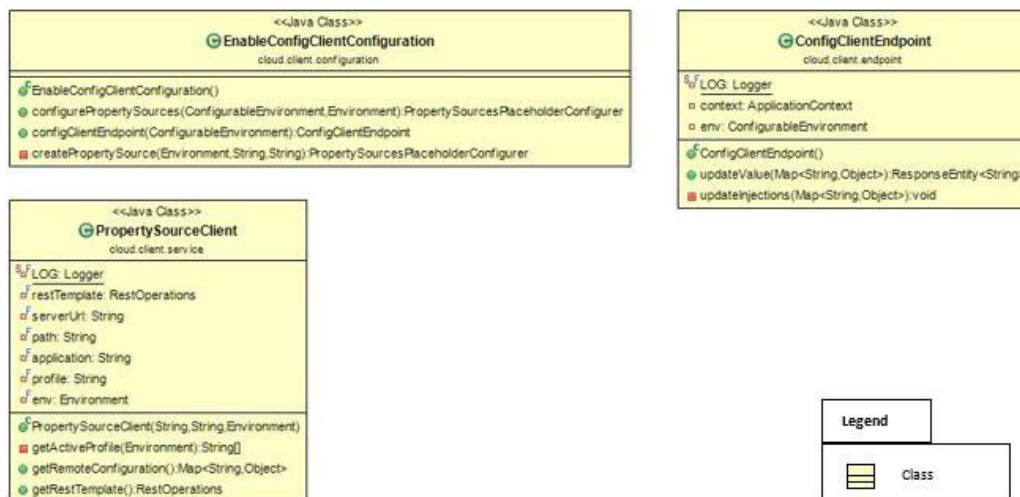


Figura 49

## Iteración 8 - Diagrama de clases del "Configuration Client"



### 4.1.3.9. Iteración 9

Esta iteración se centra en las modificaciones de las clases actuales para que soporten múltiples tenants a la vez.

La Figura 50 detalla las clases del Configuration Client mientras que la Figura 51 y Figura 52 detallan las clases del Configuration Server.

**Figura 50**

*Iteración 9 - Diagrama de clases del "Configuration Client"*

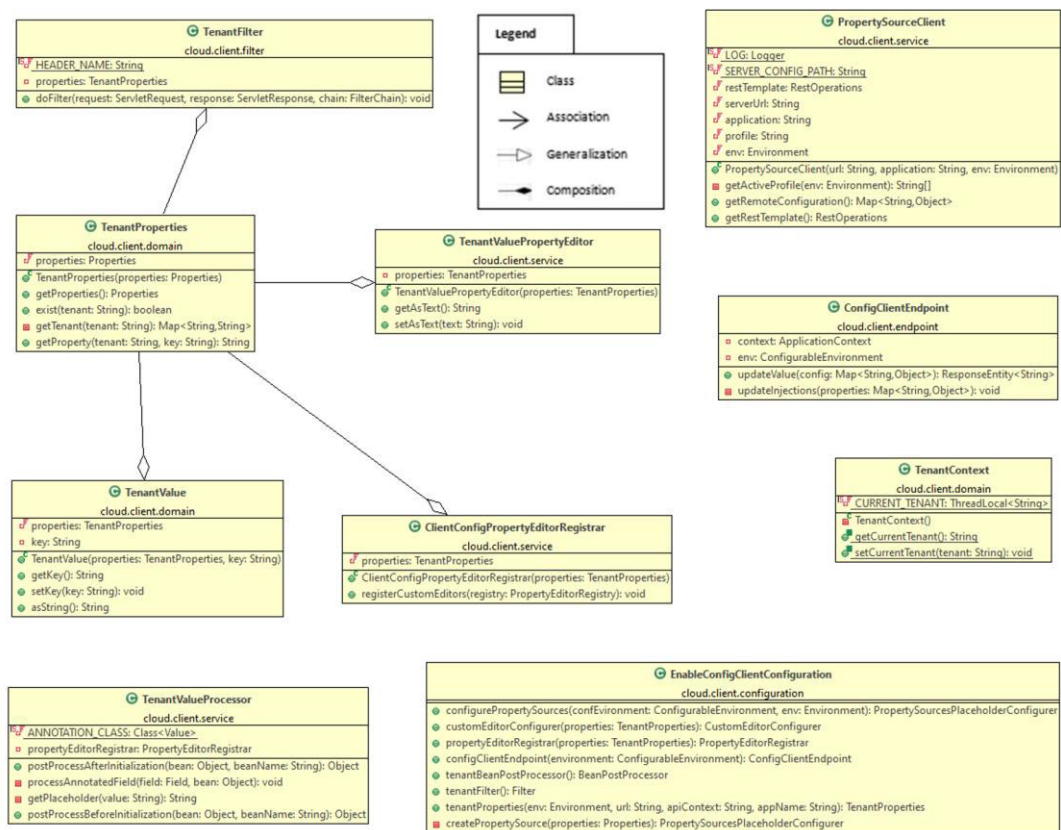


Figura 51

## Iteración 9 - Diagrama de clases del "Configuration Server" I

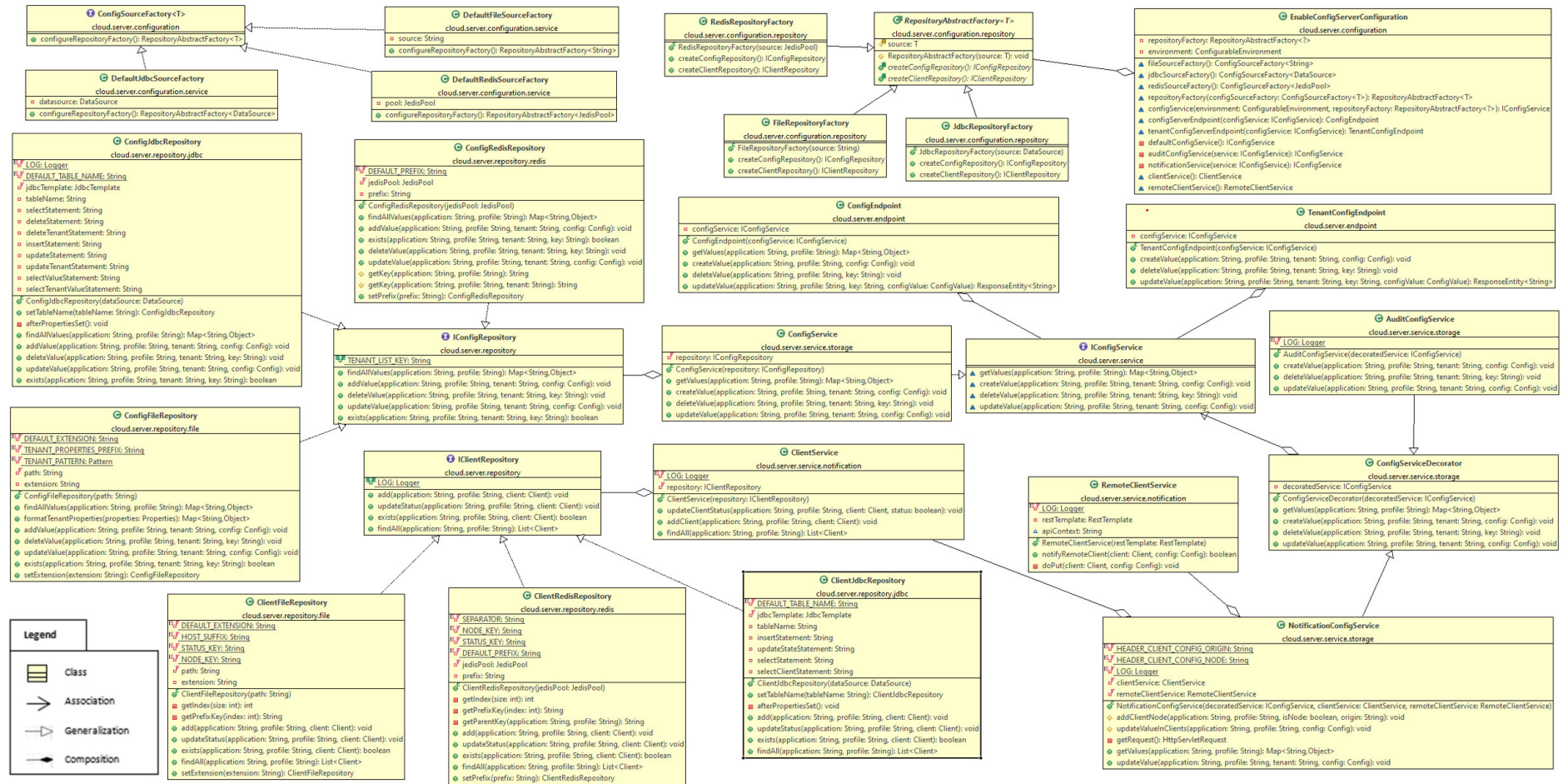
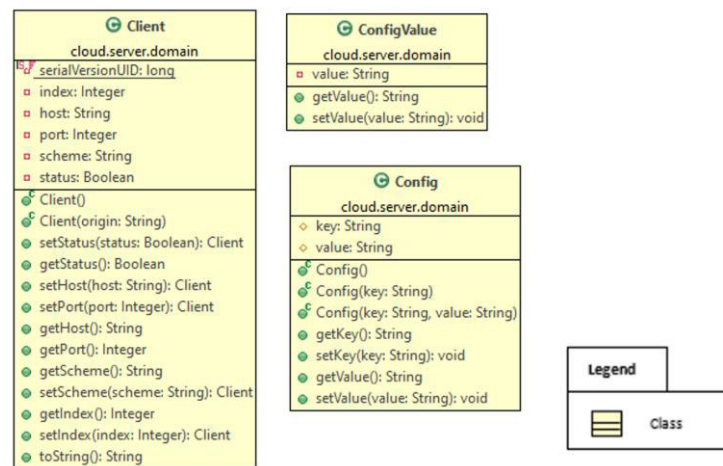


Figura 52

*Iteración 9 - Diagrama de clases del "Configuration Server" II*



#### 4.1.3.10. Iteración 10

Esta iteración se centra en las modificaciones de las clases actuales para que soporte el uso de plantillas en la generación de tenants.

En la Figura 53, Figura 54, Figura 55, Figura 56 y Figura 57 detallan todos los cambios a través del diagrama de clases.

**Figura 53**

*Iteración 10 - Diagrama de clases del "Configuration Server" I*

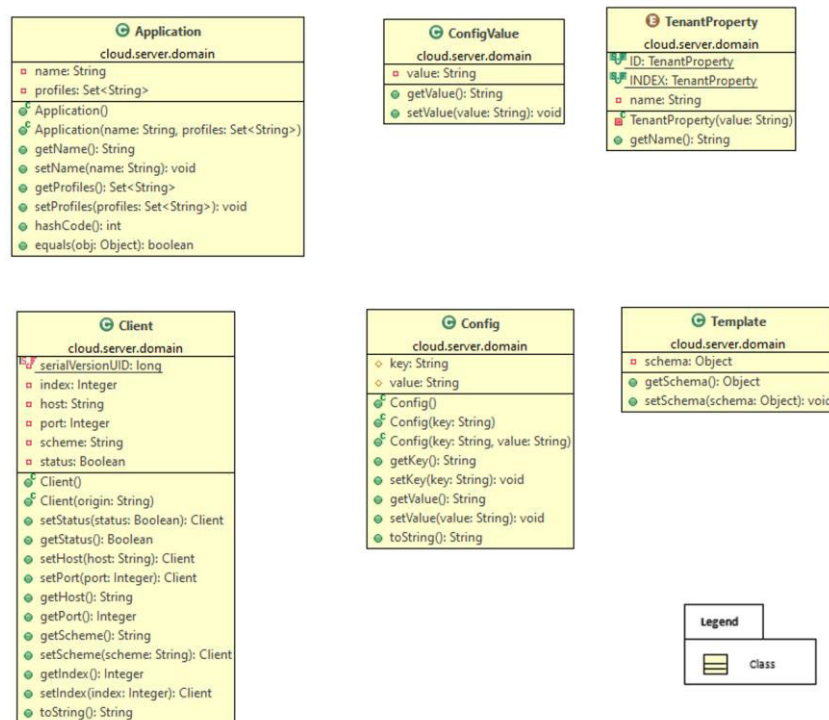


Figura 54

Iteración 10 - Diagrama de clases del "Configuration Server" II

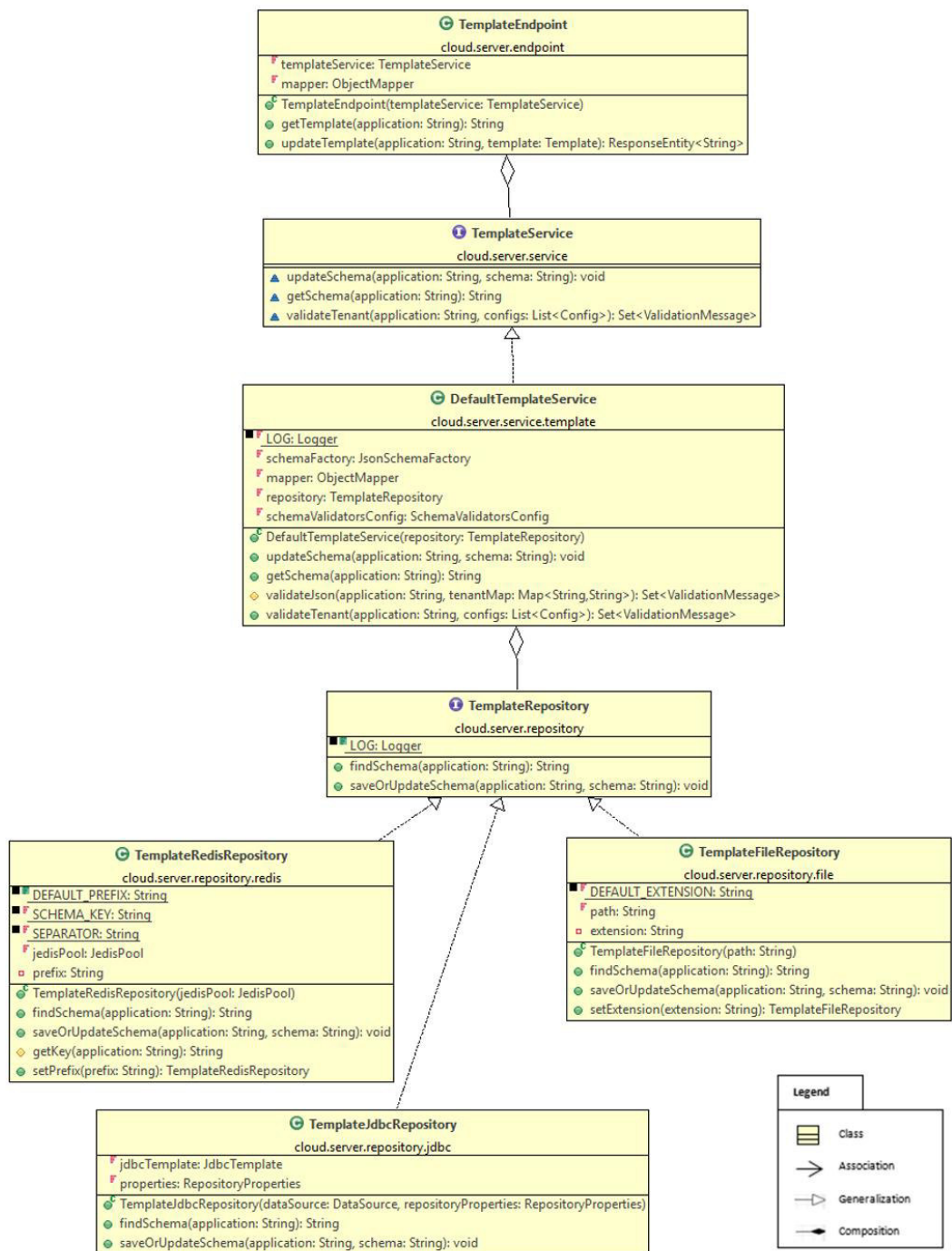


Figura 55

## Iteración 10 - Diagrama de clases del "Configuration Server" III

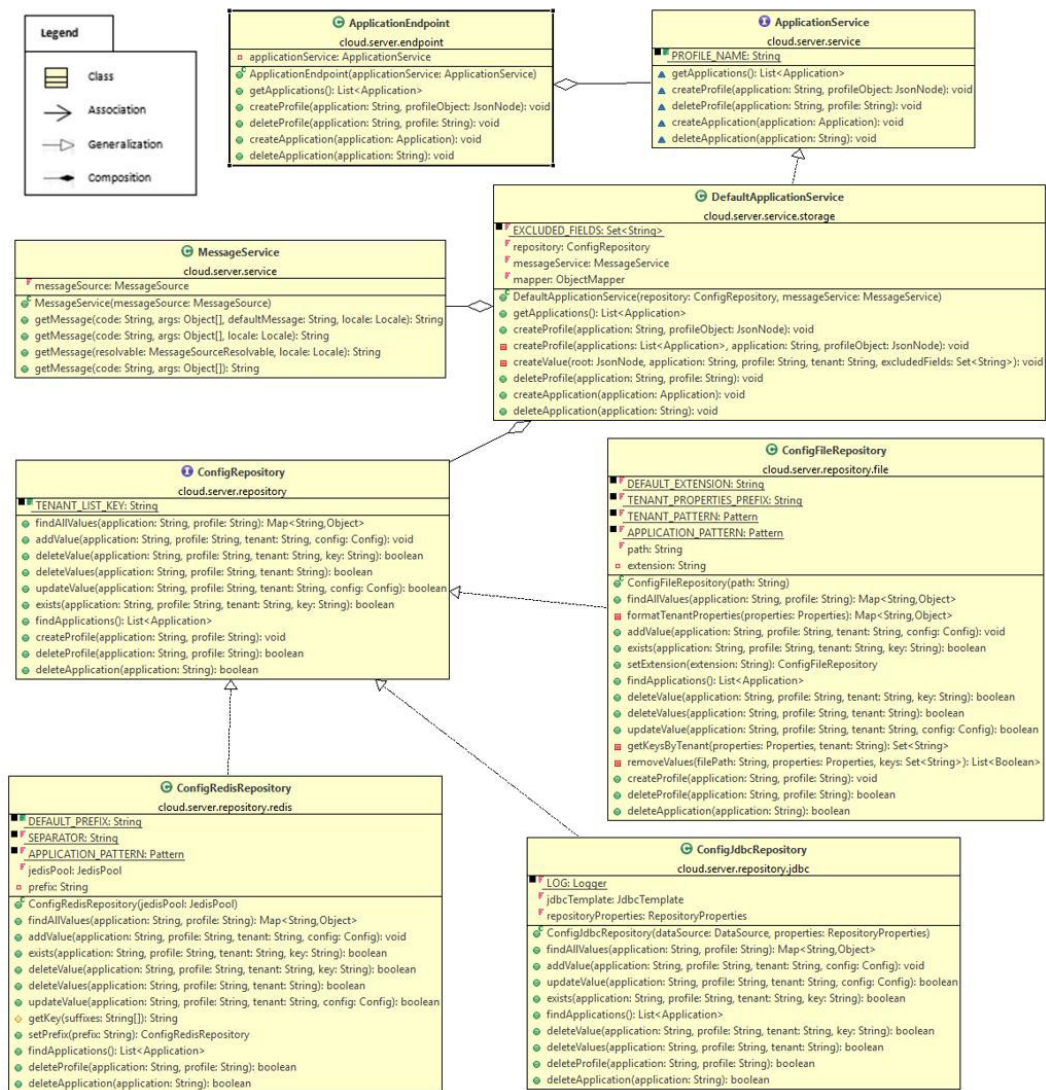


Figura 56

Iteración 10 - Diagrama de clases del "Configuration Server" IV

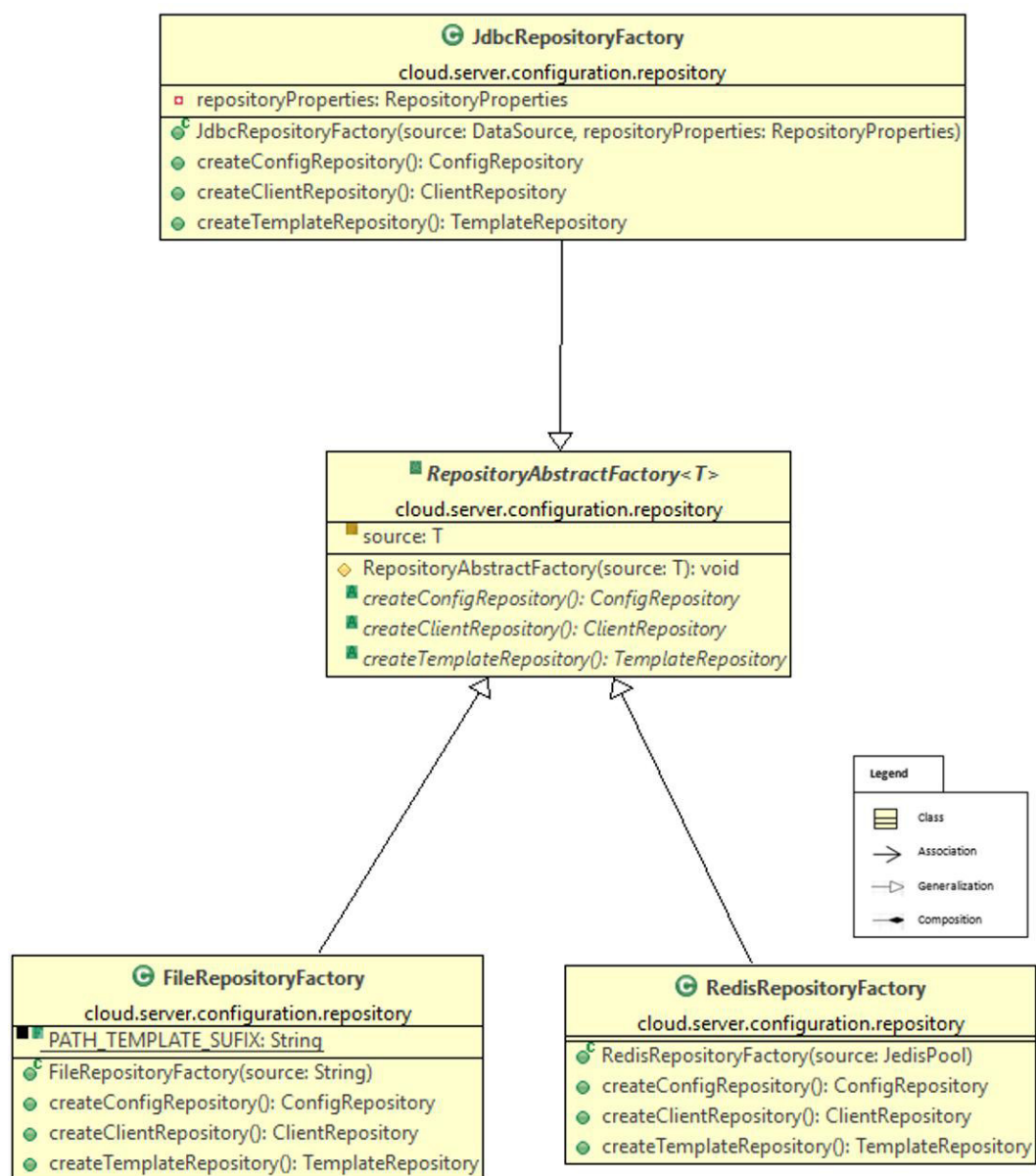


Figura 57

## Iteración 10 - Diagrama de clases del "Configuration Server" V



#### 4.1.3.11. Iteración 11

En esta iteración se refactoriza el código para hacerlo más simple, y se separan las responsabilidades según el recurso que gestiona.

En la Figura 58, Figura 59, Figura 60 y Figura 61 se muestran los cambios de la última iteración.

**Figura 58**

*Iteración 11 - Diagrama de clases del "Configuration Client"*

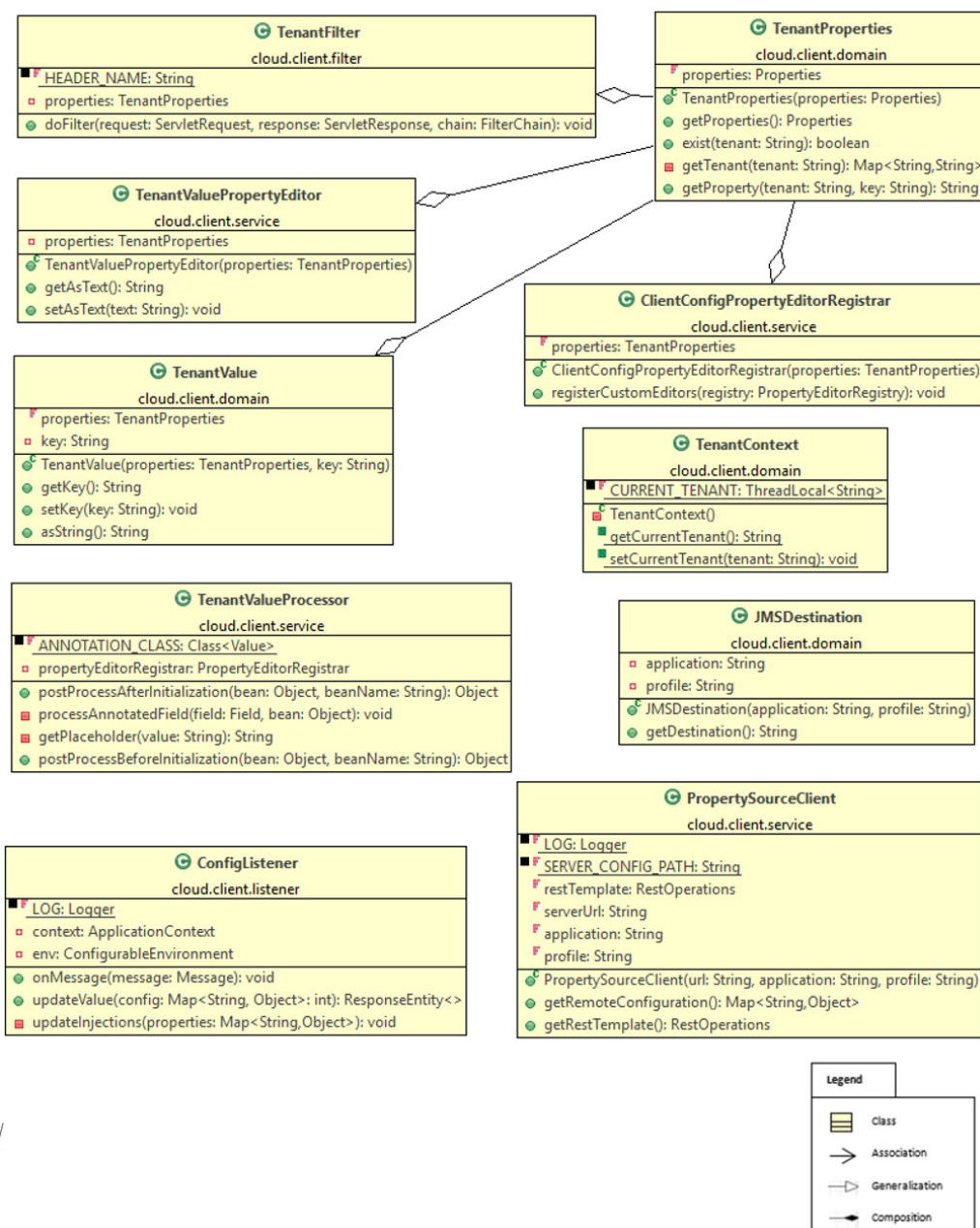


Figura 59

Iteración 11 - Diagrama de clases del "Configuration Server" I

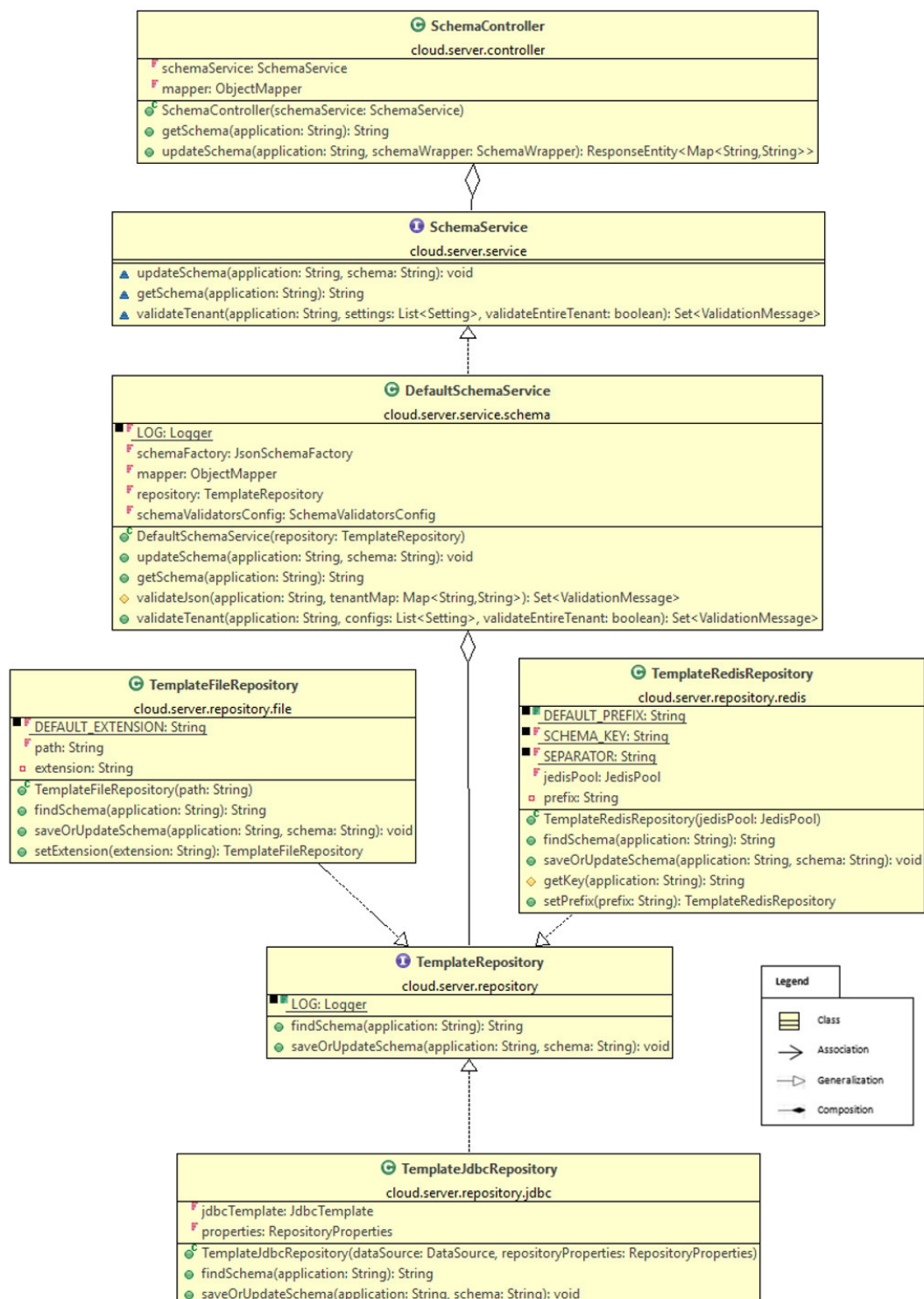
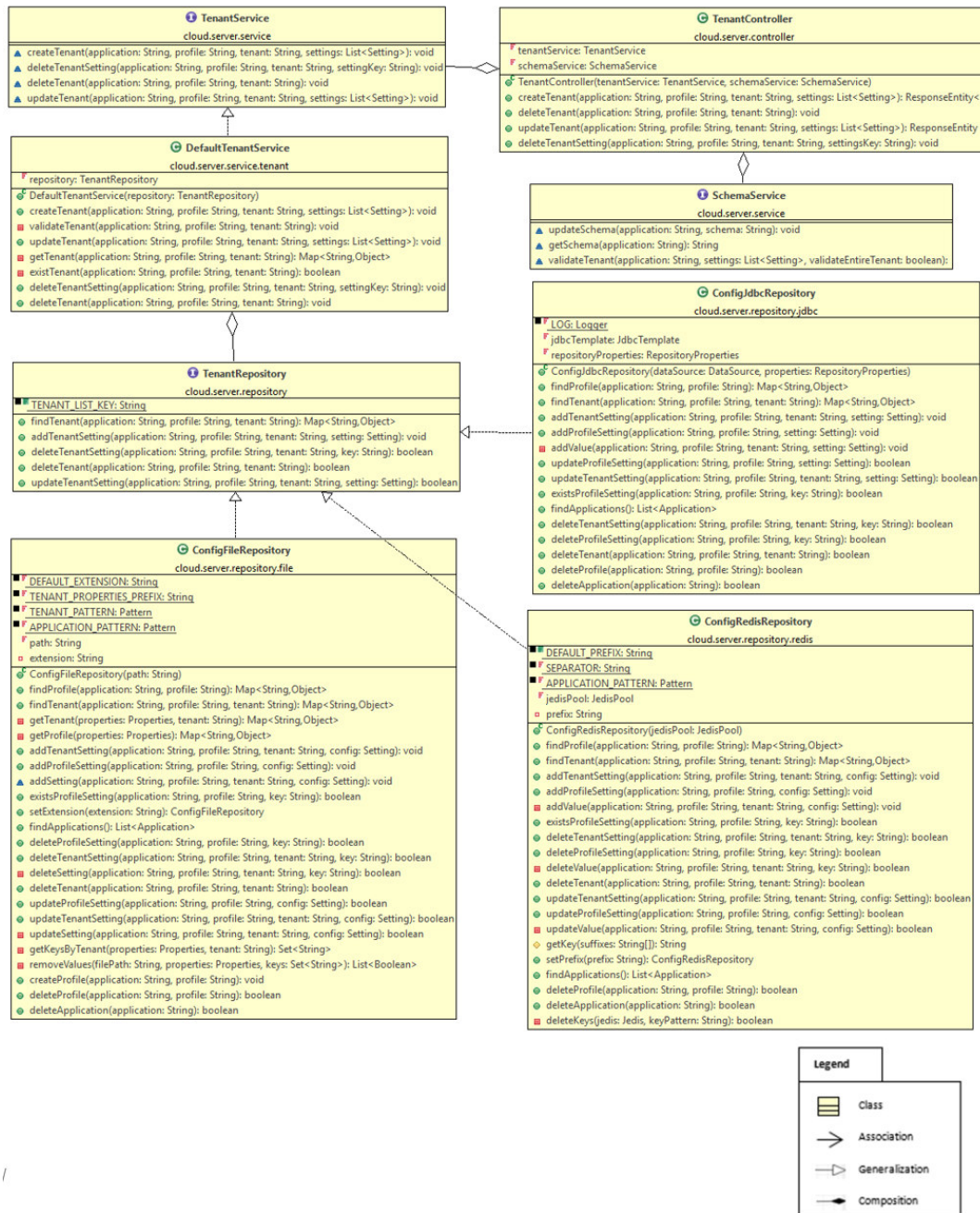


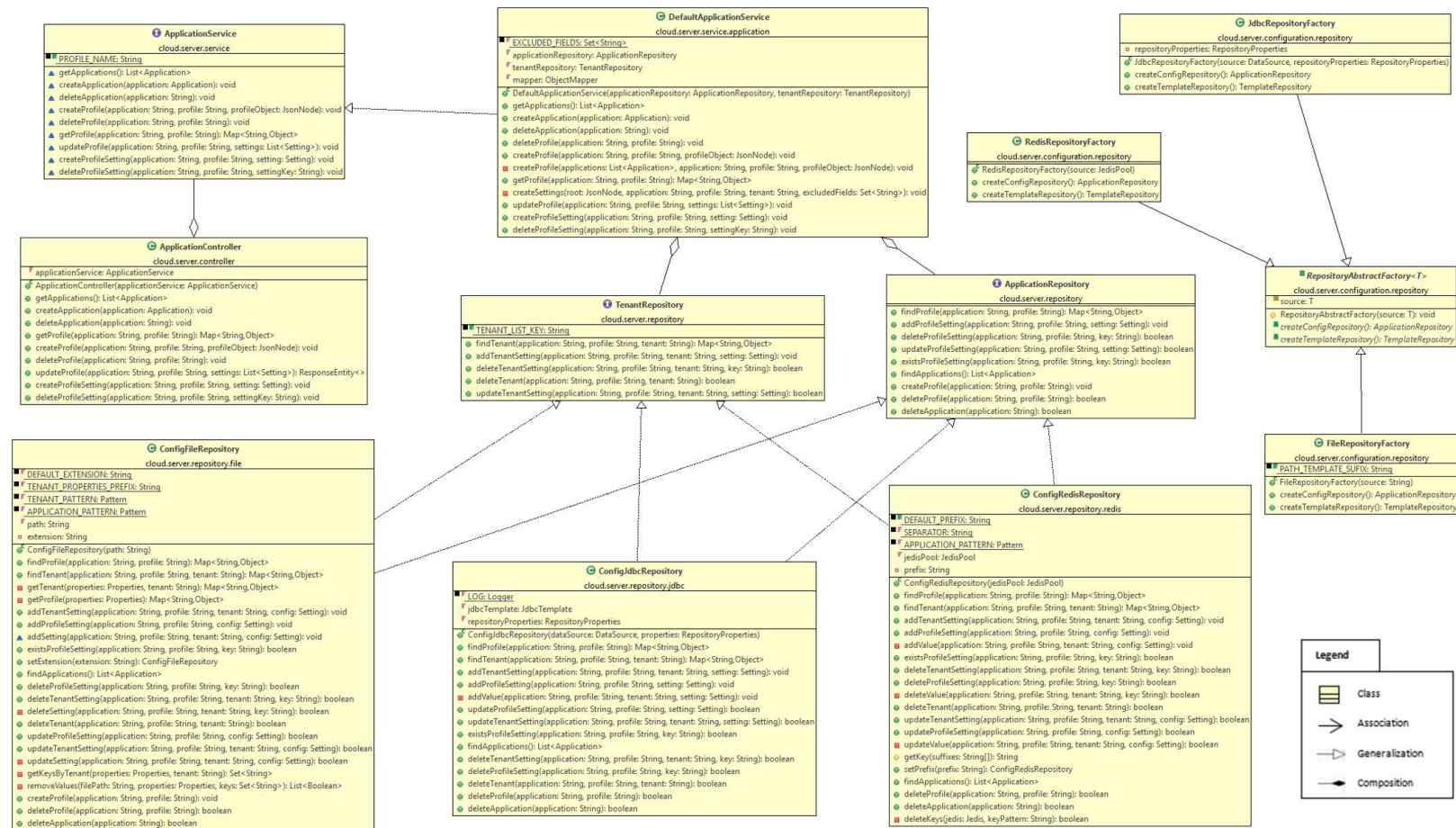
Figura 60

## Iteración 11 - Diagrama de clases del "Configuration Server" II



### Figura 61

### Iteración 11 - Diagrama de clases del "Configuration Server" III



#### 4.1.4. Fase de Puesta a Producción

En esta fase se detalla la información de la estructura de las tablas utilizadas y de las APIS expuestas.

##### 4.1.4.1. Script de Base de Datos

##### Las tablas de base de datos descritas en la Tabla 18 y

Tabla 22 deben de existir en el ambiente productivo para cuando el escenario donde el repositorio jdbc es usado.

##### Tabla 18

*Descripción de la tabla "cloud\_config\_application"*

| Columna    | Tipo      | Descripción                                                       |
|------------|-----------|-------------------------------------------------------------------|
| id         | int       | Llave primaria de la tabla.                                       |
| name       | varchar   | Nombre de la aplicación de la cual se almacenará su configuración |
| created_at | timestamp | Fecha y hora de la creación del registro.                         |

*Nota.* Se muestra la estructura de la tabla de base de datos, columnas, tipo y descripción para almacenar las aplicaciones.

##### Tabla 19

*Descripción de la tabla "cloud\_config\_profile"*

| Columna        | Tipo      | Descripción                                               |
|----------------|-----------|-----------------------------------------------------------|
| id             | int       | Llave primaria de la tabla.                               |
| name           | varchar   | Nombre del perfil del cual se almacenará su configuración |
| created_at     | timestamp | Fecha y hora de la creación del registro.                 |
| application_id | Int       | Llave foránea de la Tabla 18                              |

*Nota.* Se muestra la estructura de la tabla de base de datos, columnas, tipo y descripción para almacenar los perfiles.

**Tabla 20***Descripción de la tabla "cloud\_config\_tenant"*

| Columna    | Tipo      | Descripción                                               |
|------------|-----------|-----------------------------------------------------------|
| id         | int       | Llave primaria de la tabla.                               |
| name       | varchar   | Nombre del tenant del cual se almacenará su configuración |
| created_at | timestamp | Fecha y hora de la creación del registro.                 |
| profile_id | Int       | Llave foránea de la Tabla 19                              |

*Nota.* Se muestra la estructura de la tabla de base de datos, columnas, tipo y descripción para almacenar los tenants.

**Tabla 21***Descripción de la tabla "cloud\_config\_setting"*

| Columna    | Tipo      | Descripción                               |
|------------|-----------|-------------------------------------------|
| id         | int       | Llave primaria de la tabla.               |
| name       | varchar   | Nombre de la configuración                |
| value      | varchar   | Valor de la configuración                 |
| profile_id | Int       | Llave foránea de la Tabla 19              |
| tenant_id  | int       | Llave foránea de la Tabla 20              |
| created_at | timestamp | Fecha y hora de la creación del registro. |

*Nota.* Se muestra la estructura de la tabla de base de datos, columnas, tipo y descripción para almacenar las configuraciones.

**Tabla 22***Descripción de la tabla "cloud\_config\_schema"*

| Columna     | Tipo    | Descripción                                                    |
|-------------|---------|----------------------------------------------------------------|
| id          | int     | Llave primaria de la tabla.                                    |
| application | varchar | Nombre de la aplicación de la cual se almacenará su plantilla. |
| schema      | varchar | Json schema de la plantilla.                                   |

*Nota.* Se muestra la estructura de la tabla de base de datos, columnas, tipo y descripción para almacenar las plantillas de las aplicaciones.

#### 4.1.4.2. API REST

A continuación, se muestran las API para gestionar la configuración y plantillas almacenadas.

##### Schema

- GET /apps/{{application}}/schema
- PUT /apps/{{application}}/schema

##### Application

- GET /apps
- POST /apps
- DELETE /apps/{{application}}

##### Profiles

- GET /apps/{{application}}/profiles/{{profile}}
- POST /apps/{{application}}/profiles
- DELETE /apps/{{application}}/profiles/{{profile}}

##### Settings

- POST /apps/{{application}}/profiles/{{profile}}/settings
- PUT /apps/{{application}}/profiles/{{profile}}/settings/{{settingKey}}
- DELETE /apps/{{application}}/profiles/{{profile}}/settings/{{settingKey}}

##### Tenant

- POST /tenants/{{tenant}}
- DELETE /tenants/{{tenant}}
- DELETE /tenants/{{tenant}}/settings/{{settingKey}}
- POST /tenants/{{tenant}}/settings
- PUT /tenants/{{tenant}}/settings/{{settingKey}}

## 4.1.5. Validación del software

### 4.1.5.1. Pruebas unitarias y de integración

El desarrollo de las pruebas del framework fueron realizadas utilizando TDD. Como resultado de las pruebas del software obtenemos la métrica de cobertura de código, que mide el grado en el cual el código fuente ha sido comprobado. En la Figura 62 y Figura 63 se detallan la lista de las pruebas realizadas al **Configuración Server**.

**Figura 62**

*Lista de clases de pruebas I - Configuration Server*

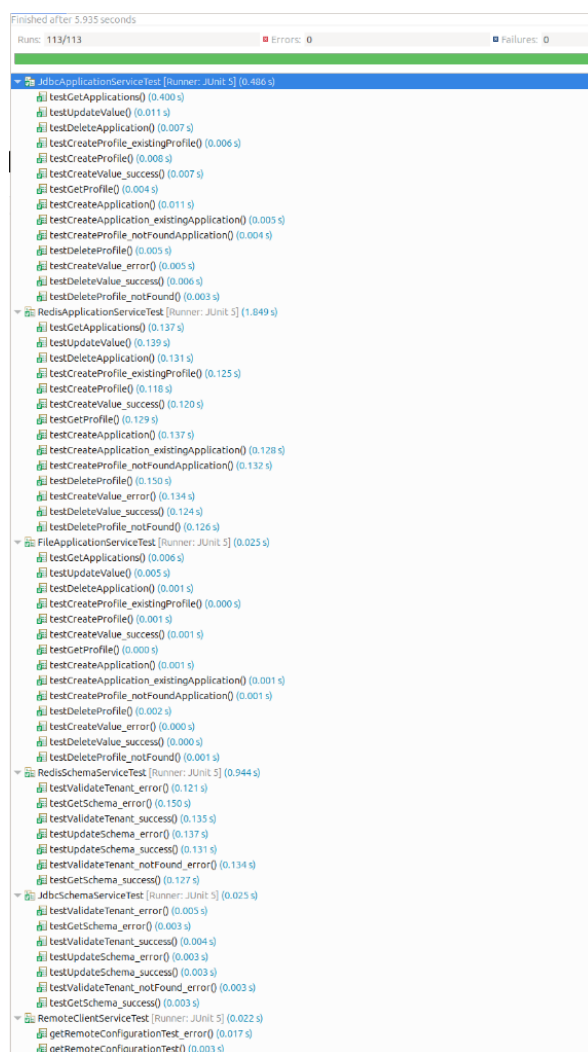


**Figura 63***Lista de clases de pruebas II - Configuration Server*

- ▼  RedisTenantServiceTest [Runner: JUnit 5] (0.963 s)
  -  testUpdateValue\_tenant() (0.129 s)
  -  testDeleteValues\_tenant\_success() (0.142 s)
  -  testCreateValue\_tenant\_success() (0.135 s)
  -  testCreateValue\_tenant\_duplicated() (0.135 s)
  -  testDeleteValue\_tenant\_success() (0.144 s)
  -  testDeleteValue\_error() (0.131 s)
  -  testGetValues\_tenants() (0.129 s)
- ▼  FileTenantServiceTest [Runner: JUnit 5] (0.021 s)
  -  testUpdateValue\_tenant() (0.011 s)
  -  testDeleteValues\_tenant\_success() (0.003 s)
  -  testCreateValue\_tenant\_success() (0.002 s)
  -  testCreateValue\_tenant\_duplicated() (0.001 s)
  -  testDeleteValue\_tenant\_success() (0.001 s)
  -  testDeleteValue\_error() (0.001 s)
  -  testGetValues\_tenants() (0.000 s)
- ▼  JdbcTenantServiceTest [Runner: JUnit 5] (0.056 s)
  -  testUpdateValue\_tenant() (0.018 s)
  -  testDeleteValues\_tenant\_success() (0.008 s)
  -  testCreateValue\_tenant\_success() (0.006 s)
  -  testCreateValue\_tenant\_duplicated() (0.006 s)
  -  testDeleteValue\_tenant\_success() (0.005 s)
  -  testDeleteValue\_error() (0.003 s)
  -  testGetValues\_tenants() (0.007 s)
- ▼  FileSchemaServiceTest [Runner: JUnit 5] (0.013 s)
  -  testValidateTenant\_error() (0.002 s)
  -  testGetSchema\_error() (0.002 s)
  -  testValidateTenant\_success() (0.002 s)
  -  testUpdateSchema\_error() (0.001 s)
  -  testUpdateSchema\_success() (0.004 s)
  -  testValidateTenant\_notFound\_error() (0.001 s)
  -  testGetSchema\_success() (0.000 s)
- ▼  ApplicationControllerTest [Runner: JUnit 5] (0.269 s)
  -  testUpdateProfileSetting() (0.079 s)
  -  testCreateProfileSetting() (0.017 s)
  -  testUpdateProfileSetting\_notFound() (0.008 s)
  -  testDeleteProfileSetting() (0.008 s)
  -  testDeleteApplication\_notFound() (0.007 s)
  -  testDeleteApplication() (0.007 s)
  -  testCreateProfile\_existingProfile() (0.008 s)
  -  testCreateProfile() (0.035 s)
  -  testGetProfile() (0.010 s)
  -  testCreateApplication() (0.020 s)
  -  testCreateApplication\_existingApplication() (0.014 s)
  -  testCreateProfile\_notFoundApplication() (0.012 s)
  -  testDeleteProfile() (0.012 s)
  -  testDeleteProfileSetting\_notFound() (0.008 s)
  -  testApplications() (0.006 s)
  -  testDeleteProfile\_notFound() (0.006 s)
- ▼  TenantControllerTest [Runner: JUnit 5] (0.058 s)
  -  testUpdateTenant\_notFound() (0.009 s)
  -  testCreateTenant() (0.007 s)
  -  testDeleteTenantSetting() (0.005 s)
  -  testCreateTenant\_duplicatedProfile() (0.007 s)
  -  testUpdateTenant() (0.007 s)
  -  testDeleteTenant() (0.006 s)
  -  testCreateTenant\_configRequired() (0.006 s)
  -  testDeleteTenantSetting\_notFound() (0.007 s)
- ▼  SchemaControllerTest [Runner: JUnit 5] (0.015 s)
  -  testUpdateValue() (0.010 s)
  -  testGetValues() (0.004 s)
- ▼  ServerConfigApplicationTests [Runner: JUnit 5] (0.000 s)
  -  contextLoads() (0.001 s)

Como resultado de la ejecución de pruebas obtenemos la cobertura del código en la Figura 64.

Figura 64

Cobertura de código - Configuration Server

| Element                               | Coverage | Covered Instructions | Missed Instructions |
|---------------------------------------|----------|----------------------|---------------------|
| server-config                         | 89.0 %   | 6,308                | 779                 |
| src/main/java                         | 83.9 %   | 3,632                | 699                 |
| cloud.server.configuration.aspect     | 31.4 %   | 55                   | 120                 |
| cloud.server.repository.util          | 69.3 %   | 199                  | 88                  |
| cloud.server.configuration            | 51.7 %   | 76                   | 71                  |
| cloud.server.service.tenant           | 71.9 %   | 166                  | 65                  |
| cloud.server.controller               | 72.7 %   | 168                  | 63                  |
| cloud.server.util                     | 80.3 %   | 257                  | 63                  |
| cloud.server.exception                | 41.8 %   | 41                   | 57                  |
| cloud.server.configuration.repository | 78.8 %   | 160                  | 43                  |
| cloud.server.service                  | 14.3 %   | 6                    | 36                  |
| cloud.server.repository.file          | 96.3 %   | 643                  | 25                  |
| cloud.server.repository.jdbc          | 96.2 %   | 602                  | 24                  |
| cloud.server.domain                   | 73.8 %   | 62                   | 22                  |
| cloud.server.service.schema           | 91.8 %   | 146                  | 13                  |
| cloud.server.repository.redis         | 98.5 %   | 607                  | 9                   |
| cloud.server.repository               | 100.0 %  | 5                    | 0                   |
| cloud.server.service.application      | 100.0 %  | 404                  | 0                   |
| cloud.server.service.notification     | 100.0 %  | 35                   | 0                   |
| src/test/java                         | 97.1 %   | 2,676                | 80                  |

En la Figura 65 y Figura 66 se detallan la lista de las pruebas unitarias realizadas y la cobertura generada por el **Configuration Client** respectivamente.

Figura 65

Lista de clases de pruebas - Configuration Client

|                                                      |           |             |
|------------------------------------------------------|-----------|-------------|
| Finished after 1.151 seconds                         |           |             |
| Runs: 6/6                                            | Errors: 0 | Failures: 0 |
| ConfigListenerTest [Runner: JUnit 5] (0.683 s)       |           |             |
| testOnMessage() (0.683 s)                            |           |             |
| PropertySourceClientTest [Runner: JUnit 5] (0.274 s) |           |             |
| getRemoteConfigurationTest_serverError() (0.241 s)   |           |             |
| getRemoteConfigurationTest() (0.032 s)               |           |             |
| TenantFilterTest [Runner: JUnit 5] (0.134 s)         |           |             |
| testDoFilter_tenantNotExist() (0.127 s)              |           |             |
| testDoFilter() (0.004 s)                             |           |             |
| testDoFilter_emptyTenant() (0.002 s)                 |           |             |

Figura 66

Cobertura de código - Configuration Client

| Element                    | Coverage | Covered Instructions | Missed Instructions |
|----------------------------|----------|----------------------|---------------------|
| client-config              | 45.7 %   | 442                  | 525                 |
| src/main/java              | 29.0 %   | 211                  | 517                 |
| cloud.client.configuration | 0.0 %    | 0                    | 207                 |
| cloud.client.service       | 35.7 %   | 90                   | 162                 |
| cloud.client.domain        | 9.1 %    | 9                    | 90                  |
| cloud.client.listener      | 48.2 %   | 54                   | 58                  |
| cloud.client.filter        | 100.0 %  | 58                   | 0                   |
| src/test/java              | 96.7 %   | 231                  | 8                   |

#### 4.1.5.2. Prueba de Concepto

Para la prueba de concepto es necesario crear dos microservicios, el primero servirá como **Configuration Server** y el segundo como **Configuration Client**, el cual consumirá las configuraciones almacenadas en el primero.

##### Configuration Server

La clase principal del **Configuration Server** cuenta con la configuración de Spring Boot y la anotación del framework como se muestra a continuación en la Figura 67.

**Figura 67**

*Configuración del "Configuration Server"*

```
@SpringBootApplication
@EnableConfigurationServer
public class Application {

    public static void main(String[] args) {
        SpringApplication.run(Application.class, args);
    }
}
```

Para la ejecución de la prueba se hace uso de un conjunto de archivos con extensión "properties" que servirán como fuente de datos para las configuraciones de las aplicaciones, para ello se tiene que configurar el application.properties del **Configuration Server**, como se muestra en la Figura 68.

La configuración dependerá de las características que se quieran habilitar como las características **push** y **log**.

## Figura 68

### *Configuración del repositorio del "Configuration Server"*

```

1  server.port = 8443
2  cloud.configuration.push=true
3  cloud.configuration.push.url=tcp://localhost:61616
4  cloud.configuration.log=true
5  cloud.configuration.source=file
6  cloud.configuration.source.file.path=/home/josem/eclipse-workspace-tesis/source/
7  cloud.configuration.api-context=
8  server.error.include-message=always

```

## Configuration Client

No necesita configuración adicional, debido a que la librería es auto configurable, es decir que solo con agregarlo como dependencia del proyecto es suficiente para su funcionamiento.

Para utilizar los valores almacenados remotamente, se utiliza la anotación `@Value` con el nombre de la configuración almacenada.

La anotación `@Refresh` en la clase y para configuraciones propias de tenants, se usa el tipo de dato `TenantValue`, como se observa en la Figura 69.

## Figura 69

### *Utilización de la anotación @Value*

```

@Refresh
@RestController
public class DemoController {

    @Value("${app-name}")
    private String nameTest;

    @Value("${web-url}")
    private TenantValue webUrl;
}

```

*Nota.* Las claves `app-name` y `web-url` representan a los nombres de las configuraciones externalizadas.

Como resultado estos atributos serán reemplazados con los valores correctos basados en que tenant realiza la petición.

## **CAPITULO 5 RESULTADOS Y DISCUSION**

En el presente capítulo, se describe la ejecución y resultado del proceso de validación desarrollado para con el framework propuesto, con el propósito de comprobar que los objetivos del trabajo de investigación fueron alcanzados.

Como primer punto, se definen los objetivos propuestos a alcanzar, continuamos describiendo y ejecutando el proceso de validación y por último la presentación de los resultados.

### **5.1. Análisis, interpretación y discusión de resultados**

En este punto se definen los objetivos de validación alineados a los objetivos del trabajo de investigación, así como la ejecución y posterior recopilación de la información para su análisis.

#### **5.1.1. Objetivos de la validación de la herramienta**

- Obtener el porcentaje de satisfacción de los usuarios al externalizar y gestionar la configuración haciendo uso del framework desarrollado.
- Obtener las métricas de los atributos de calidad del framework desarrollado.

#### **5.1.2. Diseño de la validación**

- En el proceso de validación participaron 12 profesionales entre desarrolladores y arquitectos de software, con experiencia mínima de 5 años.
- Los componentes claves de la arquitectura del framework fueron explicados a todos los participantes.
- Los participantes fueron capacitados con los procesos de configuración y utilización del framework a través de la aplicación de este en un proyecto demostrativo.

- Se realizó la encuesta Post-Uso (PSSUQ) para medir la satisfacción de uso de la herramienta. (Anexo 3 Encuesta de Satisfacción de usuario Post-Uso)
- Se configuró la herramienta SonarQube con el plugin del Modelo SQALE.
- El Framework es configurado con SonarQube para la realización del análisis estático de código.

### **5.1.3. Ejecución de la validación**

La ejecución de la validación inicio a partir de la exposición del problema sobre la gestión de la configuración en entornos multitenant, con el objetivo de obtener los diversos puntos de vista y criterios a considerar para dar solución a este problema.

Como segundo paso, se brindó una explicación de los componentes claves de la arquitectura del framework para un mejor entendimiento de la solución que brinda y la integración disponible.

Ambos pasos sobre la ejecución de las reuniones con los participantes fueron registrados como se evidencia en el Anexo 6.

También se realizó un taller práctico, el cual permitió configurar el framework desde cero y utilizarlo sobre el proyecto que fue preparado con anterioridad como se describe en el Anexo 7, el proyecto realizado en el ide eclipse simula un proyecto existente que requiere utilizar el framework propuesto.

La documentación realizada para la configuración y funcionamiento del framework fueron también proveídos durante el taller como se muestra en el Anexo.

### 5.1.3.1. Encuesta Post-Uso

Como último paso del taller, Todos los participantes fueron parte de la encuesta Post-Uso (Anexo 3 Encuesta de Satisfacción de usuario Post-Uso), con el objetivo de medir la satisfacción de los mimos sobre la herramienta presentada y los resultados se muestran en la Tabla 23.

**Tabla 23**

*Resultado de la encuesta de satisfacción de usuario*

|                                  | Cuestionario |    |    |    |    |    |    |    |    |     |     |     |     |     |     |        |
|----------------------------------|--------------|----|----|----|----|----|----|----|----|-----|-----|-----|-----|-----|-----|--------|
|                                  | C1           | C2 | C3 | C4 | C5 | C6 | C7 | C8 | C9 | C10 | C11 | C12 | C13 | C14 | C15 | C16    |
| P1                               | 1            | 1  | 1  | 1  | 1  | 2  | 2  | 2  | 1  | 2   | 1   | 1   | 1   | 2   | 2   | 1      |
| P2                               | 2            | 2  | 2  | 2  | 2  | 2  | 7  | 3  | 2  | 2   | 2   | 2   | 3   | 4   | 2   | 2      |
| P3                               | 1            | 1  | 1  | 1  | 1  | 2  | 1  | 1  | 2  | 2   | 2   | 1   | 1   | 1   | 2   | 1      |
| P4                               | 3            | 3  | 2  | 2  | 2  | 2  | 3  | 3  | 4  | 3   | 2   | 2   | 2   | 2   | 3   | 2      |
| P5                               | 1            | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1   | 1   | 1   | 1   | 1   | 1   | 1      |
| P6                               | 1            | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1  | 1   | 1   | 1   | 1   | 1   | 1   | 1      |
| P7                               | 2            | 3  | 3  | 2  | 3  | 2  | 2  | 1  | 3  | 2   | 3   | 3   | 2   | 2   | 2   | 2      |
| P8                               | 1            | 1  | 1  | 1  | 1  | 1  | 2  | 2  | 2  | 2   | 2   | 2   | 1   | 2   | 1   | 1      |
| P9                               | 1            | 1  | 1  | 2  | 1  | 2  | 1  | 1  | 3  | 2   | 2   | 2   | 2   | 2   | 1   | 1      |
| P10                              | 1            | 2  | 2  | 1  | 1  | 2  | 2  | 2  | 2  | 2   | 2   | 2   | 1   | 1   | 1   | 1      |
| P11                              | 1            | 1  | 2  | 1  | 1  | 1  | 1  | 1  | 1  | 1   | 3   | 1   | 2   | 2   | 2   | 1      |
| P12                              | 1            | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2  | 2   | 2   | 2   | 2   | 2   | 2   | 2      |
| Sumatoria de varianzas           |              |    |    |    |    |    |    |    |    |     |     |     |     |     |     | 8,979  |
| Varianza de la suma de los ítems |              |    |    |    |    |    |    |    |    |     |     |     |     |     |     | 69,243 |

Con los valores obtenidos en la Tabla 23 se procedió a realizar la validación de confiabilidad haciendo uso de la variable Alfa de Cronbach en la Tabla 24.

**Tabla 24**

*Alfa de Cronbach*

| Alfa de Cronbach | Cantidad de participantes |
|------------------|---------------------------|
| 0.93             | 12                        |

El resultado de la confiabilidad es **excelente**, garantizando los resultados obtenidos en la encuesta Post-Uso.

Con el objetivo de tener una mejor interpretación de los resultados, la Tabla 25 muestra los datos a través de sus subescalas del PSSUQ.

**Tabla 25**

*Resultados de la encuesta por escala*

| Puntuación General<br>PSSUQ | Utilidad del Sistema<br>(SYSUSE) | Calidad de la Información<br>(INFOQUAL) | Calidad de la Interfaz<br>(INTERQUAL) |
|-----------------------------|----------------------------------|-----------------------------------------|---------------------------------------|
| 1.38                        | 1.17                             | 1.50                                    | 1.50                                  |
| 2.56                        | 2.00                             | 3.00                                    | 2.75                                  |
| 1.31                        | 1.17                             | 1.50                                    | 1.25                                  |
| 2.50                        | 2.33                             | 2.83                                    | 2.25                                  |
| 1.00                        | 1.00                             | 1.00                                    | 1.00                                  |
| 1.00                        | 1.00                             | 1.00                                    | 1.00                                  |
| 2.31                        | 2.50                             | 2.33                                    | 2.00                                  |
| 1.44                        | 1.00                             | 2.00                                    | 1.25                                  |
| 1.56                        | 1.33                             | 1.83                                    | 1.50                                  |
| 1.56                        | 1.50                             | 2.00                                    | 1.00                                  |
| 1.38                        | 1.17                             | 1.33                                    | 1.75                                  |
| 1.94                        | 1.83                             | 2.00                                    | 2.00                                  |
| 1.66                        | 1.50                             | 1.86                                    | 1.60                                  |

La escala de Utilidad del Sistema (SYSUSE) se obtuvo promediando las respuestas de las preguntas 1 a 6, mientras que la de Calidad de Información (INFOQUAL) se calculó con el promedio de las preguntas 7 a 12.

Finalmente, la escala de Calidad de la Interfaz (INTERQUAL) se basó en el promedio de las respuestas a las preguntas 13 a 16.

Tomando como referencia PSSUQ, se obtiene que los resultados fueron altamente favorables. La puntuación total promedio fue de 1.66, lo cual representa una percepción de usabilidad significativamente superior a la media reportada por Lewis (2002), que es de 2.82. Asimismo, las subescalas reflejan resultados similares como se muestra en la Tabla 26.

**Tabla 26**

*Comparación con valores normativos*

| Subescala   | Resultado | Valor Promedio |
|-------------|-----------|----------------|
| SYSUSE      | 1.5       | 2.80           |
| INFOQUAL    | 1.86      | 3.02           |
| INTERQUAL   | 1.6       | 2.49           |
| PSSUQ Total | 1.66      | 2.82           |

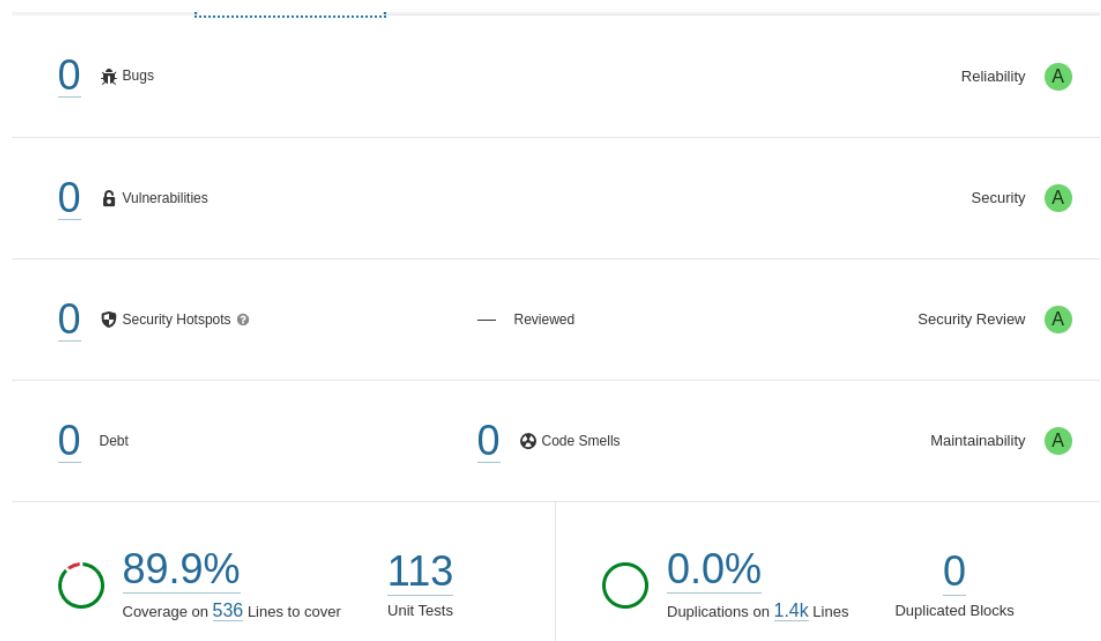
Dado que en el PSSUQ una menor puntuación indica mayor satisfacción, estos resultados permiten concluir que el sistema evaluado ofrece una experiencia de usuario considerablemente mejor que la media de referencia establecida por Lewis (2002).

5.1.3.2.      **Análisis de código estático**

Finalmente, en otro entorno, se realizó el análisis del código estático del framework, para ello se hizo uso de la herramienta Sonarqube, el cual como resultado generó métricas de los atributos de calidad a evaluar, como se observa en la Figura 70.

**Figura 70**

Métricas sonarqube del framework propuesto



También se utilizó el método SQALE para determinar la calidad técnica del software, que como se observa en la Figura 71, se obtiene la escala A, que según la calificación SQALE, el framework cumple con altos estándares de calidad en lo que refiere a código limpio, legible y bien estructurado, también con métricas de complejidad, acoplamiento y cohesión dentro los limites ideales para su fácil entendimiento y modificación.

Figura 71

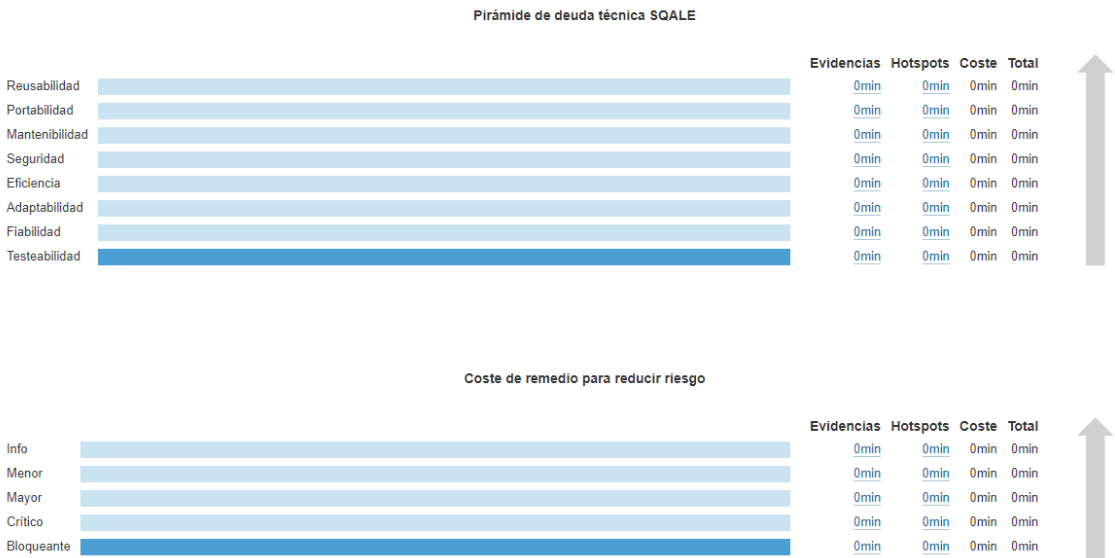
Calificación SQALE



Tanto la pirámide de deuda técnica como el coste para reducir el riesgo del proyecto es 0, debido a la escala obtenida por el framework, lo que indica que no se necesitan realizar cambios como se observa en la Figura 72.

Figura 72

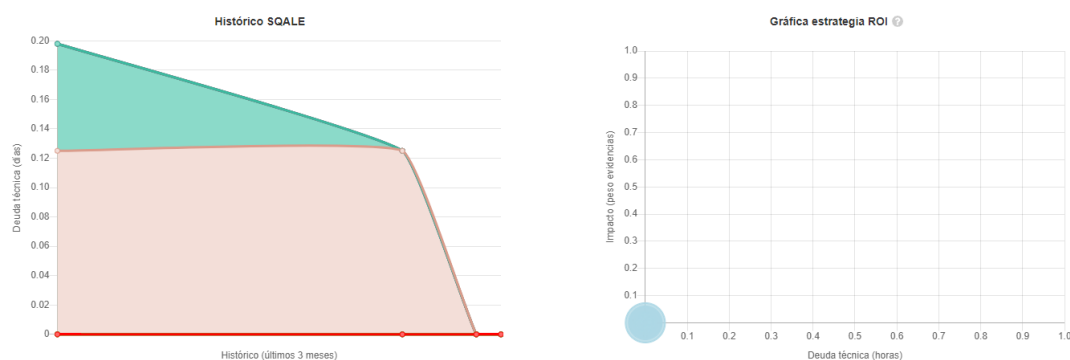
Pirámide de deuda técnica y coste para reducir el riesgo



En la Figura 73, la gráfica de burbuja nos ayuda a saber que componentes se debe mejorar para obtener el mayor retorno de inversión basándose en la deuda técnica y el rubro del negocio. Pudiéndose observar, que no se necesitan mejoras tampoco.

**Figura 73**

*Histórico SQALE y estrategia ROI*



Adicionalmente, para evaluar el rendimiento del framework se hizo uso de la herramienta JProfiler, con el objetivo de analizar el comportamiento del framework en cuanto a rendimiento se refiere.

## 5.2. Pruebas de hipótesis

En esta sección se probarán las hipótesis descritas en la Matriz de Consistencia (Anexo 2) que son de interés para el trabajo de investigación.

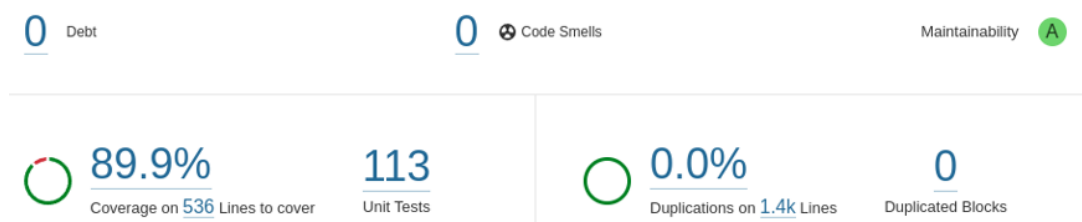
### 5.2.1. Prueba de hipótesis del objetivo específico 1

**HOE1** El framework “config-manager” con un alto nivel de mantenibilidad para externalizar la configuración en entornos multitenant.

El nivel **A** alcanzado por el **atributo de mantenibilidad** de la herramienta sonarqube como se observa en la Figura 74, demuestra que el código es altamente mantenible, contando con baja deuda técnica, cero code smells y una complejidad manejable.

**Figura 74**

*Métricas sonarqube - Mantenibilidad*



Para obtener el nivel de mantenibilidad. Sonarqube evaluó internamente diversas métricas, como la deuda técnica, code smells, complejidad ciclomatica, duplicación de código, cobertura del código, tamaño del código, entre otros.

### 5.2.2. Prueba de hipótesis del objetivo específico 2

**HOE2** El framework “config-manager” con un alto nivel de fiabilidad para externalizar la configuración en entornos multitenant.

Como resultado del análisis estático del código se obtuvo también el nivel **A** para el **atributo de fiabilidad** y cero bugs como se muestra en la Figura 75, el cual sumado a la alta cobertura del código, garantizan el funcionamiento consistente del framework y el cumplimiento de las expectativas del usuario siendo altamente fiable.

**Figura 75**

*Métricas sonarqube - Fiabilidad*



Para determinar el nivel asignado, la herramienta evalúa diversas métricas como la cantidad de bugs, excepciones no controladas, errores de concurrencia, desbordamiento de pilas, condiciones de error, entre otros durante el análisis del código.

### 5.2.3. Prueba de hipótesis del objetivo específico 3

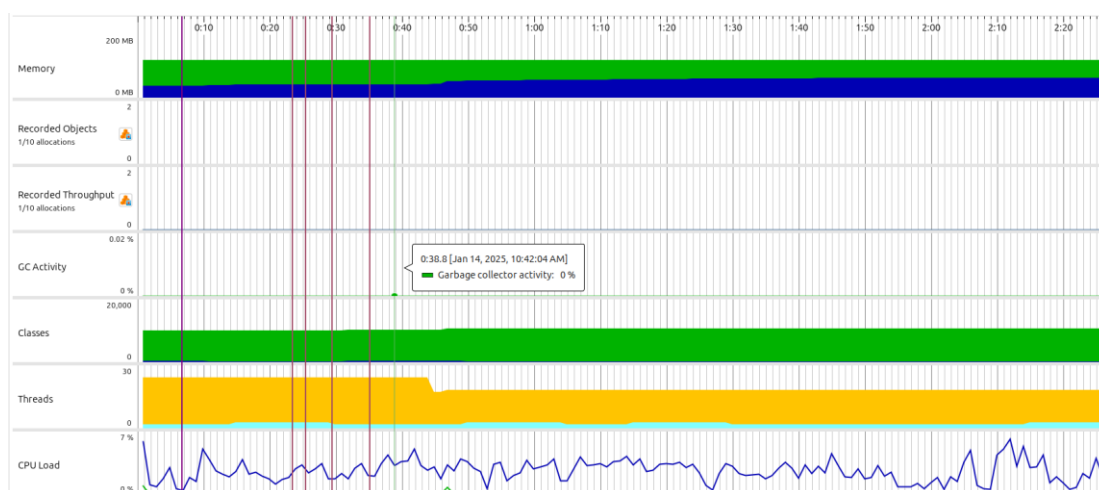
**HOE3** El framework “config-manager” con un alto nivel de eficiencia para externalizar la configuración en entornos multitenant.

Como se observa en la Figura 76, durante el uso del framework , el consumo de memoria se mantuvo estable y a niveles mínimos, además no encontrarse fugas de esta.

El consumo del CPU también es estable, entre 0 y 7% y no se detectaron cuellos de botella que puedan afectar el rendimiento.

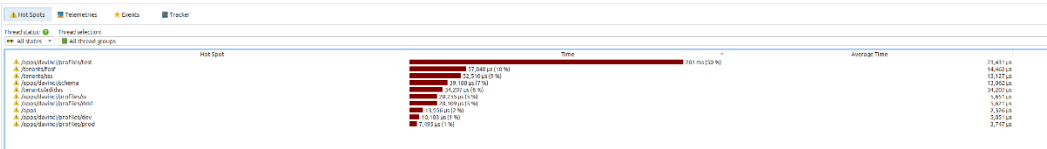
**Figura 76**

*Resultado JProfiler - Recursos*



Por otro lado, como se observa en la Figura 77, los tiempos de respuesta de las APIs fueron menores a 50 milisegundos, a excepción de una que fue menor a los 200 milisegundos, considerándose los resultados como respuestas rápidas y aceptables respectivamente

**Figura 77**  
*Resultado JProfiler - APIs*



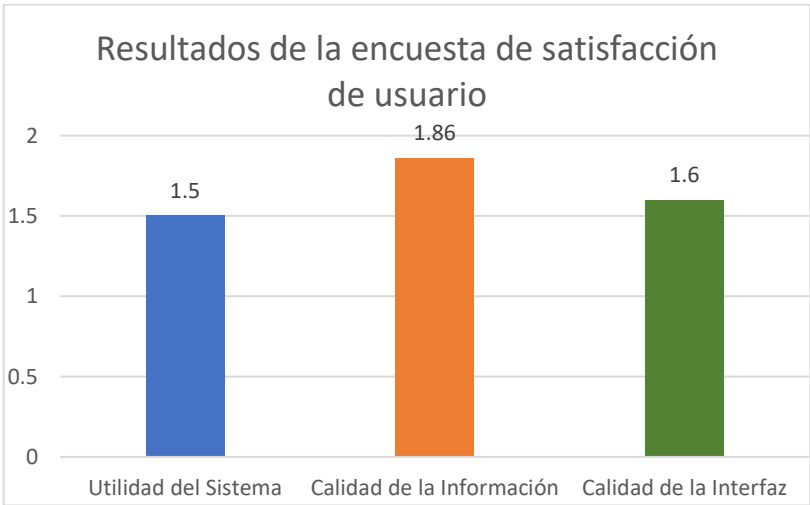
Tanto la medición de los recursos como la de rendimiento, expresados en el bajo consumo de memoria, CPU, y tiempo de respuesta demuestran la eficiencia del framework.

**5.2.4. Prueba de hipótesis del objetivo específico 4**

**HOE4** El framework “config-manager” con un alto nivel de satisfacción del usuario para externalizar la configuración en entornos multitenant.

La medición de la satisfacción del usuario como se observa en la Figura 78, demuestran se generó una alta satisfacción en los usuarios, debido a que los resultados de las escalas varían entre 1.5 y 1.86, siendo cercanos a 1, lo que indica una buena usabilidad, facilidad de uso y satisfacción con el sistema según PSSUQ.

**Figura 78**  
*Diagrama de barras del resultado de la encuesta por escalas*



### 5.3. Presentación de resultados

Se verifica que el framework cuenta con un alto nivel de mantenibilidad al demostrarse la HOE1 a través de la métrica de mantenibilidad donde se obtuvo 0% de deuda técnica como se observa en la Figura 74

Se verifica que el framework cuenta con un alto nivel de fiabilidad al demostrarse la HOE2 a través de la métrica de fiabilidad obtenida en la Figura 75, donde se observa que se cuenta con cero bugs y 89.9% de cobertura de código.

Se verifica el alto nivel de eficiencia del framework al demostrarse la HOE3 a través de las mediciones del uso recursos y rendimiento, de donde se obtuvo un bajo consumo de recursos y alto rendimiento como se observa en la Figura 76 y la Figura 77 respectivamente.

Se verifica el alto nivel de satisfacción del usuario para con el framework al demostrarse la HOE4 a través de la encuesta PSSUQ donde se obtuvo una puntuación general de 1.66 como se observa en la Tabla 25 .

## CONCLUSIONES

Se alcanzó el objetivo principal de la tesis que consistió en implementar el framework “config-manager” para externalizar la configuración en entornos multitenant, en el capítulo 3 numeral 3.9 se definió la propuesta solución. Mientras que en el capítulo 4 se detalló el proceso de implementación desde la definición de las historias de usuario hasta la validación del artefacto.

Se logró implementar el framework “config-manager” con un alto nivel de mantenibilidad para externalizar la configuración en entornos multitenant, aplicando buenas prácticas de desarrollo de software, como el uso de estándares de codificación, revisiones de código, refactorización y patrones de diseño, lo cual permitió obtener 0% de deuda técnica.

Se logró implementar el framework “config-manager” con un alto nivel de fiabilidad para externalizar la configuración en entornos multitenant a través de la aplicación de un desarrollo guiado por pruebas (TDD) y de la creación de diversos tipos de pruebas automatizadas, como las unitarias, de integración y resiliencia, lo cual generó 89.9% de cobertura de código, que facilitó la reducción de la cantidad de bugs.

Se logró implementar el framework “config-manager” con un alto nivel de eficiencia para externalizar la configuración en entornos multitenant, haciendo uso de procesos paralelos y asíncronos, y optimizando los algoritmos para evitar bucles innecesarios, permitiendo optimizar el uso de los recursos del sistema, lo cual permitió obtener tiempos de respuesta menores a los 200 milisegundos.

Se logró implementar el framework “config-manager” con un alto nivel de satisfacción del usuario para externalizar la configuración en entornos multitenant, utilizando como referencia la encuesta de satisfacción post-uso (PSSUQ) de donde se obtuvo una puntuación de 1.66 que se traduce en una mayor satisfacción por parte de los participantes de la encuesta.

## RECOMENDACIONES

1. Con el objetivo de tener el framework más seguro, se recomienda adicionar los mecanismos de seguridad respectivos, como la autenticación y encriptación, que permita el intercambio seguro de los datos entre los clientes y el servidor de la configuración.
2. Para ampliar la funcionalidad del framework, se recomienda realizar las implementaciones que permitan integrar otras herramientas de almacenes de datos, como AWS, Vault, entre otras, para que las aplicaciones obtengan todas las configuraciones de un servidor centralizado.
3. Se recomienda crear los spring boot starters para el framework para reducir el número de dependencias que se agregan manualmente en los proyectos clientes.
4. Como complemento se puede mejorar el aspecto de auditoria para registrar más información cuando cualquier petición es procesada en el framework.

## REFERENCIAS BIBLIOGRAFICAS

- Aditi, S., & Parmeet, K. (2023). A Survey of Distributed Data Storage in the Cloud for Multitenant Applications. *International Journal of Performability Engineering*, 19(3), 184. <https://doi.org/10.23940/ijpe.23.03.p4.184192>
- Balalaie, A., Heydarnoori, A., & Jamshidi, P. (2016). Microservices Architecture Enables DevOps: Migration to a Cloud-Native Architecture. *IEEE Software*, 33(3), 42-52. <https://doi.org/10.1109/MS.2016.64>
- Bayramcavus, A., Kaya, M. C., & Dogru, A. H. (2021). Interoperability of Microservice-Based Systems. *2021 13th International Conference on Electrical and Electronics Engineering (ELECO)*, 594-598. <https://doi.org/10.23919/ELECO54474.2021.9677712>
- Bien, N. H., & Dan Thu, T. (2015). Multi-tenant web application framework architecture pattern. *2015 2nd National Foundation for Science and Technology Development Conference on Information and Computer Science (NICS)*, 40-48. <https://doi.org/10.1109/NICS.2015.7302219>
- ckittel. (2023). *External Configuration Store pattern—Azure Architecture Center*. <https://learn.microsoft.com/en-us/azure/architecture/patterns/external-configuration-store>
- Etcd*. (2023). Etcd. <https://etcd.io/>
- Hall, G. M. (2010). Data Access Layer. En G. M. Hall, *Pro WPF and Silverlight MVVM* (pp. 163-183). Apress. [https://doi.org/10.1007/978-1-4302-3163-9\\_8](https://doi.org/10.1007/978-1-4302-3163-9_8)
- Hernández Sampieri, R., & Fernández-Collado, C. F. (2014). *Metodología de la investigación* (P. Baptista Lucio, Ed.; Sexta edición). McGraw-Hill Education.
- Hofmann, M., Schnabel, E., & Stanley, K. (2016). *Microservices Best Practices for Java*.

- Hofmann, M., Schnabel, E., & Stanley, K. (2017). *Microservices Best Practices for Java*.
- Huamán Rojas, J. A., Treviños Noa, L. L., & Medina Flores, W. A. (2022). Epistemología de las investigaciones cuantitativas y cualitativas. *Horizonte de la Ciencia*, 12(23).  
<https://doi.org/10.26490/uncp.horizonteciencia.2022.23.1462>
- Ji, S., Tang, Z., Wang, W., Li, H., Yao, J., & Jacobsen, H.-A. (2024). Ripple: Large-Scale Service and Configuration Management in the Cloud. *Proceedings of the 25th International Middleware Conference*, 354-366.  
<https://doi.org/10.1145/3652892.3700777>
- Kale, S. S., & Borhade, R. H. (2013). Development of Multitenant SaaS framework at single instance and with zero effort multitenancy. *2013 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, 834-839. <https://doi.org/10.1109/ICACCI.2013.6637284>
- Karatas, G., Can, F., Dogan, G., Konca, C., & Akbulut, A. (2017). Multi-tenant architectures in the cloud: A systematic mapping study. *2017 International Artificial Intelligence and Data Processing Symposium (IDAP)*, 1-4.  
<https://doi.org/10.1109/IDAP.2017.8090268>
- Lee, E., Kwon, K., & Yun, J. (2020). Performance Measurement of GraphQL API in Home ESS Data Server. *2020 International Conference on Information and Communication Technology Convergence (ICTC)*, 1929-1931.  
<https://doi.org/10.1109/ICTC49870.2020.9289569>
- Lewis, J. R. (2002). Psychometric Evaluation of the PSSUQ Using Data from Five Years of Usability Studies. *International Journal of Human-Computer Interaction*, 14(3-4), 463-488. <https://doi.org/10.1080/10447318.2002.9669130>

- Lomotey, R. K., & Deters, R. (2013). Reliable Consumption of Web Services in a Mobile-Cloud Ecosystem Using REST. *2013 IEEE Seventh International Symposium on Service-Oriented System Engineering*, 13-24. <https://doi.org/10.1109/SOSE.2013.10>
- Matharu, G. S., Mishra, A., Singh, H., & Upadhyay, P. (2015). Empirical Study of Agile Software Development Methodologies: A Comparative Analysis. *ACM SIGSOFT Software Engineering Notes*, 40(1), 1-6. <https://doi.org/10.1145/2693208.2693233>
- mcleanbyron. (2023, marzo 29). *What is Azure App Configuration?* <https://learn.microsoft.com/en-us/azure/azure-app-configuration/overview>
- Microservices Pattern: Externalized configuration*. (2023). [microservices.io](http://microservices.io/patterns/externalized-configuration.html). <http://microservices.io/patterns/externalized-configuration.html>
- Netflix/archaius*. (2023). [Java]. Netflix, Inc. <https://github.com/Netflix/archaius> (Obra original publicada en 2012)
- Odun-Ayo, I., Misra, S., Abayomi-Alli, O., & Ajayi, O. (2017). Cloud Multi-Tenancy: Issues and Developments. *Companion Proceedings of The 10th International Conference on Utility and Cloud Computing*, 209-214. <https://doi.org/10.1145/3147234.3148095>
- Ouh, E. L., & Kok Siew Gan, B. (2023). An Exploratory Study of Architectural Style and Effort Estimation for Multi-Tenant Microservices-Based Software as a Service (SaaS). *2023 IEEE 20th International Conference on Software Architecture Companion (ICSA-C)*, 159-166. <https://doi.org/10.1109/ICSA-C57050.2023.00043>
- Sayagh, M., Dong, Z., Andrzejak, A., & Adams, B. (2017). Does the Choice of Configuration Framework Matter for Developers? Empirical Study on 11 Java

- Configuration Frameworks. *2017 IEEE 17th International Working Conference on Source Code Analysis and Manipulation (SCAM)*, 41-50. <https://doi.org/10.1109/SCAM.2017.25>
- Sharma, A., & Kaur, P. (2019). A Multitenant Data Store Using a Column Based NoSQL Database. *2019 Twelfth International Conference on Contemporary Computing (IC3)*, 1-5. <https://doi.org/10.1109/IC3.2019.8844906>
- Sharma, A., & Kaur, P. (2021). *Implementing a Multitenant System using a Document-based NoSQL Database*.
- Spring Cloud Config*. (2023). <https://docs.spring.io/spring-cloud-config/docs/current/reference/html/>
- Tang, E., & Fan, Y. (2016). Performance Comparison between Five NoSQL Databases. *2016 7th International Conference on Cloud Computing and Big Data (CCBD)*, 105-109. <https://doi.org/10.1109/CCBD.2016.030>
- What Is AWS AppConfig? - AWS AppConfig*. (2023). <https://docs.aws.amazon.com/appconfig/latest/userguide/what-is-appconfig.html>
- What is Consul? | Consul | HashiCorp Developer*. (2023). <https://developer.hashicorp.com/consul/docs/intro>
- What is Multi-Tenant | IBM*. (2023). <https://www.ibm.com/topics/multi-tenant>
- Yucra Quispe, T., & Bernedo Villalta, L. Z. (2020). Epistemología e Investigación Cuantitativa. *IGOBERNANZA*, 3(12), 107-120. <https://doi.org/10.47865/igob.vol3.2020.88>
- ZooKeeper: Because Coordinating Distributed Systems is a Zoo*. (2023). <https://zookeeper.apache.org/doc/r3.8.1/zookeeperOver.html>

## ANEXOS

### Anexo 1 Operacionalización de las variables

| Variables                                                              | Tipos de variables | Definición de variables                                      | Dimensiones o factores   | Medición de indicador o cumplimiento                                       | Definición de indicador                            |
|------------------------------------------------------------------------|--------------------|--------------------------------------------------------------|--------------------------|----------------------------------------------------------------------------|----------------------------------------------------|
| Framework de desarrollo                                                | Independiente      | Herramienta con API REST                                     | Mantenibilidad           | Capacidad para ser modificado debido a necesidades de corrección o mejora. | Bajo o inexistente porcentaje de deuda técnica.    |
|                                                                        |                    |                                                              | Fiabilidad               | Capacidad para desempeñar sus funciones de manera correcta.                | Baja o inexistente cantidad de bugs encontrados.   |
|                                                                        |                    |                                                              | Eficiencia               | % de utilización de recursos de hardware.                                  | Cantidad de recursos utilizados para su ejecución. |
| Gestión y Externalización de la configuración en entornos multitenant. | Dependiente        | Externalización de la configuración en entornos multitenant. | Satisfacción del usuario | # de satisfacción.                                                         | Alto porcentaje de satisfacción del usuario.       |

## Anexo 2 Matriz de Consistencia

| Problema                                                                                                                                                                                              | Objetivos                                                                                                                                                                    | Hipótesis                                                                                                                                                         | Metodología                                                                                                             |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------|
| <b>Problema General</b><br>¿De qué manera se puede implementar el framework “config-manager” para externalizar la configuración en entornos multitenant?                                              | <b>Objetivo General</b><br>Implementar el framework “config-manager” para externalizar la configuración en entornos multitenant.                                             | <b>Hipótesis General</b><br>El framework “config-manager” para externalizar la configuración en entornos multitenant.                                             | <b>Metodología de Investigación</b><br><b>Tipo de Investigación</b><br>Aplicada<br><br><b>Enfoque:</b><br>Cuantitativo. |
| <b>Problemas Específicos</b><br><br>¿De qué manera se puede implementar el framework “config-manager” con un alto nivel de mantenibilidad para externalizar la configuración en entornos multitenant? | <b>Objetivos Específico</b><br><br>Implementar el framework “config-manager” con un alto nivel de mantenibilidad para externalizar la configuración en entornos multitenant. | <b>Hipótesis Específicas</b><br><br>El framework “config-manager” con un alto nivel de mantenibilidad para externalizar la configuración en entornos multitenant. | <b>Recolección de Datos:</b><br>Encuestas.                                                                              |
| ¿De qué manera se puede implementar el framework “config-manager” con un alto nivel de fiabilidad para externalizar la configuración en entornos multitenant?                                         | Implementar el framework “config-manager” con un alto nivel de fiabilidad para externalizar la configuración en entornos multitenant.                                        | El framework “config-manager” con un alto nivel de fiabilidad para externalizar la configuración en entornos multitenant.                                         | <b>Diseño:</b><br>No experimental.                                                                                      |
| ¿De qué manera se puede implementar el framework “config-manager” con un alto nivel de eficiencia para externalizar la configuración en entornos multitenant?                                         | Implementar el framework “config-manager” con alto nivel de eficiencia para externalizar la configuración en entornos multitenant.                                           | El framework “config-manager” con un alto nivel de eficiencia para externalizar la configuración en entornos multitenant.                                         | <b>Alcances o estrategias:</b><br>Correlacional                                                                         |
| ¿De qué manera se puede implementar el framework “config-manager” con un alto nivel de satisfacción del usuario para externalizar la configuración en entornos multitenant?                           | Implementar el framework “config-manager” con alto nivel de satisfacción del usuario para externalizar la configuración en entornos multitenant.                             | El framework “config-manager” con un alto nivel de satisfacción del usuario para externalizar la configuración en entornos multitenant.                           | <b>Población:</b><br>Profesionales de Tecnología de Información relacionados activamente con el desarrollo de software. |
|                                                                                                                                                                                                       |                                                                                                                                                                              |                                                                                                                                                                   | <b>Muestra:</b><br>12 profesionales.                                                                                    |
|                                                                                                                                                                                                       |                                                                                                                                                                              |                                                                                                                                                                   | <b>Tipo de Muestreo:</b><br>No probabilístico.                                                                          |

### Anexo 3 Encuesta de Satisfacción de usuario Post-Uso

| Nro<br>. | Cuestionario de usabilidad del sistema<br>posterior al estudio<br>Versión 3                                                   | Totalmente de<br>acuerdo |   |   |   | Totalmente en<br>desacuerdo |   |   | NA |
|----------|-------------------------------------------------------------------------------------------------------------------------------|--------------------------|---|---|---|-----------------------------|---|---|----|
|          |                                                                                                                               | 1                        | 2 | 3 | 4 | 5                           | 6 | 7 |    |
| 1        | En general, estoy satisfecho con lo fácil que es usar la herramienta.                                                         |                          |   |   |   |                             |   |   |    |
| 2        | Fue sencillo utilizar esta herramienta.                                                                                       |                          |   |   |   |                             |   |   |    |
| 3        | Fui capaz de completar las tareas y escenarios rápidamente usando esta herramienta.                                           |                          |   |   |   |                             |   |   |    |
| 4        | Me sentí cómodo usando esta herramienta.                                                                                      |                          |   |   |   |                             |   |   |    |
| 5        | Fue fácil aprender a usar esta herramienta.                                                                                   |                          |   |   |   |                             |   |   |    |
| 6        | Creo que podría volverme productivo rápidamente usando esta herramienta.                                                      |                          |   |   |   |                             |   |   |    |
| 7        | La herramienta dió mensajes de error que claramente me decían como solucionar problemas                                       |                          |   |   |   |                             |   |   |    |
| 8        | Cada vez que cometía un error al usar la herramienta, podía recuperarme fácil y rápidamente                                   |                          |   |   |   |                             |   |   |    |
| 9        | La información (como ayuda en línea, mensajes en pantalla y otra documentación) proporcionada con esta herramienta fue clara. |                          |   |   |   |                             |   |   |    |
| 10       | Fue fácil encontrar la información que necesitaba.                                                                            |                          |   |   |   |                             |   |   |    |
| 11       | La información fue efectiva para ayudarme a completar las tareas y escenarios.                                                |                          |   |   |   |                             |   |   |    |
| 12       | La organización de la información en la herramienta fue clara.                                                                |                          |   |   |   |                             |   |   |    |
| 13       | La interfaz de esta herramienta fue agradable                                                                                 |                          |   |   |   |                             |   |   |    |
| 14       | Me gustó usar la interfaz de esta herramienta.                                                                                |                          |   |   |   |                             |   |   |    |
| 15       | Esta herramienta web tiene todas las funciones y capacidades que espero que tenga.                                            |                          |   |   |   |                             |   |   |    |
| 16       | En general, estoy satisfecho con esta herramienta.                                                                            |                          |   |   |   |                             |   |   |    |

**Nota.** Adaptado de “Psychometric Evaluation of the PSSUQ Using Data from Five Years of Usability Studies”, por Lewis, 2002.

# Anexo 4 Historias de Usuario

Tabla 27

*Historia de usuario 5*

| Historia de Usuario                                                                                      |                             |
|----------------------------------------------------------------------------------------------------------|-----------------------------|
| Número: 5                                                                                                | Usuario: Aplicación Cliente |
| Nombre de Historia: Registrar el historial de cambios de todos los eventos.                              |                             |
| Prioridad:                                                                                               | Riesgo de desarrollo: Media |
| Puntos estimados:                                                                                        | Iteración asignada: 5       |
| Programador responsable: José Mori                                                                       |                             |
| Descripción:                                                                                             |                             |
| Todas las operaciones realizadas sobre los datos almacenados son registradas en el log de la aplicación. |                             |
| Criterios de aceptación                                                                                  |                             |
| El framework registra todos los eventos en el log de la aplicación.                                      |                             |
| Los registros del log contienen la aplicación, el perfil y la operación realizada.                       |                             |
| El usuario puede deshabilitar en cualquier momento esta funcionalidad.                                   |                             |
| Observaciones:                                                                                           |                             |
| Se trabajará bajo el marco de trabajo de SLF4J                                                           |                             |

**Tabla 28***Historia de usuario 6*

| Historia de Usuario                                                                                         |                                    |
|-------------------------------------------------------------------------------------------------------------|------------------------------------|
| <b>Número:</b> 6                                                                                            | <b>Usuario:</b> Aplicación Cliente |
| <b>Nombre de Historia:</b> Facilitar la configuración y uso del framework para la aplicación cliente.       |                                    |
| <b>Prioridad:</b>                                                                                           | <b>Riesgo de desarrollo:</b> Media |
| <b>Puntos estimados:</b>                                                                                    | <b>Iteración asignada:</b> 6       |
| <b>Programador responsable:</b> José Mori                                                                   |                                    |
| <b>Descripción:</b>                                                                                         |                                    |
| Con el objetivo de facilitar la configuración, se implementan anotaciones para su configuración automática. |                                    |
| <b>Criterios de aceptación</b>                                                                              |                                    |
| El framework permite la configuración del módulo cliente y servidor a través de anotaciones.                |                                    |
| <b>Observaciones:</b>                                                                                       |                                    |

**Tabla 29***Historia de usuario 7*

| Historia de Usuario                                                                                                                                       |                                    |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------|
| <b>Número:</b> 7                                                                                                                                          | <b>Usuario:</b> Aplicación Cliente |
| <b>Nombre de Historia:</b> Permitir el almacenamiento local de la configuración en la aplicación.                                                         |                                    |
| <b>Prioridad:</b>                                                                                                                                         | <b>Riesgo de desarrollo:</b> Baja  |
| <b>Puntos estimados:</b>                                                                                                                                  | <b>Iteración asignada:</b> 7       |
| <b>Programador responsable:</b> José Mori                                                                                                                 |                                    |
| <b>Descripción:</b>                                                                                                                                       |                                    |
| Se consultan los valores almacenados del proyecto y se cargan en el contexto de la aplicación cliente de forma automática para que puedan ser utilizados. |                                    |
| <b>Criterios de aceptación</b>                                                                                                                            |                                    |
| El framework puede consumir la configuración de una aplicación y perfil a través de un endpoint.                                                          |                                    |
| El framework carga la configuración consumida en el contexto de Spring automáticamente.                                                                   |                                    |
| <b>Observaciones:</b>                                                                                                                                     |                                    |
| La consulta de los valores almacenados se realiza con el id de la aplicación y el perfil de esta.                                                         |                                    |
| La configuración de conexión con el servidor de configuración es registrada en un archivo (properties o yaml)                                             |                                    |

**Tabla 30***Historia de usuario 8*

| Historia de Usuario                                                                                                                                                                                                                                                                                        |                                    |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------|
| <b>Número:</b> 8                                                                                                                                                                                                                                                                                           | <b>Usuario:</b> Aplicación Cliente |
| <b>Nombre de Historia:</b> Permitir la actualización de la configuración en tiempo real.                                                                                                                                                                                                                   |                                    |
| <b>Prioridad:</b>                                                                                                                                                                                                                                                                                          | <b>Riesgo de desarrollo:</b> Media |
| <b>Puntos estimados:</b>                                                                                                                                                                                                                                                                                   | <b>Iteración asignada:</b> 8       |
| <b>Programador responsable:</b> José Mori                                                                                                                                                                                                                                                                  |                                    |
| <b>Descripción:</b>                                                                                                                                                                                                                                                                                        |                                    |
| <p>Cuando los valores de la configuración son actualizados en el servidor, los valores de los clientes también deben ser refrescados.</p>                                                                                                                                                                  |                                    |
| <b>Criterios de aceptación</b>                                                                                                                                                                                                                                                                             |                                    |
| <p>El usuario puede establecer la conexión con la cola a interactuar.</p> <p>El framework debe permitir el envío de mensajes cada vez que los valores almacenados son modificados.</p> <p>El framework debe permitir la actualización de la configuración de las aplicaciones clientes en tiempo real.</p> |                                    |
| <b>Observaciones:</b>                                                                                                                                                                                                                                                                                      |                                    |
| <p>Se debe de contar con una cola para realizar la actualización en tiempo real.</p> <p>La cola debe ser configurada en el proyecto.</p>                                                                                                                                                                   |                                    |

Tabla 31

*Historia de usuario 9*

| Historia de Usuario                                                                                                                                                                                             |                                    |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------|
| <b>Número:</b> 9                                                                                                                                                                                                | <b>Usuario:</b> Aplicación Cliente |
| <b>Nombre de Historia:</b> Gestionar la configuración del proyecto a través de tenants.                                                                                                                         |                                    |
| <b>Prioridad:</b>                                                                                                                                                                                               | <b>Riesgo de desarrollo:</b> Media |
| <b>Puntos estimados:</b>                                                                                                                                                                                        | <b>Iteración asignada:</b> 9       |
| <b>Programador responsable:</b> José Mori                                                                                                                                                                       |                                    |
| <b>Descripción:</b>                                                                                                                                                                                             |                                    |
| Los valores almacenados podrán ser gestionados de acuerdo con la aplicación, perfil, y tenant al que pertenecen. Donde el tenant es opcional ya que aún se puede almacenar información por aplicación y perfil. |                                    |
| <b>Criterios de aceptación</b>                                                                                                                                                                                  |                                    |
| La información almacenada es gestionada a través de una estructura de aplicaciones, perfiles y tenants.                                                                                                         |                                    |
| El cliente puede manipular la información almacenada usando el nombre de la aplicación, perfil y tenant.                                                                                                        |                                    |
| <b>Observaciones:</b>                                                                                                                                                                                           |                                    |
| La consulta de los valores se realiza con el id de la aplicación, perfil y tenant;                                                                                                                              |                                    |

Tabla 32

*Historia de usuario 10*

| Historia de Usuario                                                                                                   |                                    |
|-----------------------------------------------------------------------------------------------------------------------|------------------------------------|
| <b>Número:</b> 10                                                                                                     | <b>Usuario:</b> Aplicación Cliente |
| <b>Nombre de Historia:</b> Permitir la creación de plantillas para dinamizar la creación de tenants.                  |                                    |
| <b>Prioridad:</b>                                                                                                     | <b>Riesgo de desarrollo:</b> Media |
| <b>Puntos estimados:</b>                                                                                              | <b>Iteración asignada:</b> 10      |
| <b>Programador responsable:</b> José Mori                                                                             |                                    |
| <b>Descripción:</b>                                                                                                   |                                    |
| Los tenants pueden ser creados a partir de una plantilla que almacenan la estructura de la información.               |                                    |
| <b>Criterios de aceptación</b>                                                                                        |                                    |
| El usuario puede manipular las plantillas a través de una API REST.                                                   |                                    |
| Cada Proyecto puede contener solo una plantilla. Y el framework valida la información almacenada usando la plantilla. |                                    |
| <b>Observaciones:</b>                                                                                                 |                                    |

Tabla 33

Historia de usuario 11

| Historia de Usuario                                                                                     |                             |
|---------------------------------------------------------------------------------------------------------|-----------------------------|
| Número: 11                                                                                              | Usuario: Aplicación Cliente |
| Nombre de Historia: Facilitar el consumo de APIs y personalización de servicios.                        |                             |
| Prioridad:                                                                                              | Riesgo de desarrollo: Alta  |
| Puntos estimados:                                                                                       | Iteración asignada: 11      |
| Programador responsable: José Mori                                                                      |                             |
| Descripción:                                                                                            |                             |
| Simplificar las APIs y refactorizar los servicios con el objetivo de cumplir una función.               |                             |
| Criterios de aceptación                                                                                 |                             |
| Los endpoints son más claros y siguen las buenas prácticas de diseño.                                   |                             |
| El código debe ser más claro, modular y fácil de entender.                                              |                             |
| Después de la refactorización, los servicios siguen funcionando igual y entregan los mismos resultados. |                             |
| Observaciones:                                                                                          |                             |

## Anexo 5 Tareas de Programación

**Tabla 34**

*Tarea de programación 5*

| Tarea                                                                                                                   |                             |
|-------------------------------------------------------------------------------------------------------------------------|-----------------------------|
| <b>Número de tarea:</b> 5                                                                                               | <b>Número historia:</b> 3   |
| <b>Nombre de tarea:</b> Servicios de gestión de valores de configuración en Redis.                                      |                             |
| <b>Tipo de tarea:</b> Desarrollo                                                                                        | <b>Puntos estimados:</b> 21 |
| <b>Fecha inicio:</b>                                                                                                    | <b>Fecha fin:</b>           |
| <b>Programador responsable:</b> José Mori                                                                               |                             |
| <b>Descripción:</b>                                                                                                     |                             |
| Se crea el conjunto de servicios (bien estructurados y extensibles) que permita la consulta de los valores almacenados. |                             |

**Tabla 35**

*Tarea de programación 6*

| Tarea                                                                                              |                            |
|----------------------------------------------------------------------------------------------------|----------------------------|
| <b>Número de tarea:</b> 6                                                                          | <b>Número historia:</b> 4  |
| <b>Nombre de tarea:</b> Soporte para múltiples perfiles.                                           |                            |
| <b>Tipo de tarea:</b> Desarrollo                                                                   | <b>Puntos estimados:</b> 5 |
| <b>Fecha inicio:</b>                                                                               | <b>Fecha fin:</b>          |
| <b>Programador responsable:</b> José Mori                                                          |                            |
| <b>Descripción:</b>                                                                                |                            |
| Se agregará la capacidad de soportar múltiples perfiles para el mismo conjunto de configuraciones. |                            |

Tabla 36

*Tarea de programación 7*

| Tarea                                                        |                     |
|--------------------------------------------------------------|---------------------|
| Número de tarea: 7                                           | Número historia: 4  |
| Nombre de tarea: Soporte para múltiples aplicaciones.        |                     |
| Tipo de tarea: Desarrollo                                    | Puntos estimados: 5 |
| Fecha inicio:                                                | Fecha fin:          |
| Programador responsable: José Mori                           |                     |
| Descripción:                                                 |                     |
| Se agregará la capacidad de soportar múltiples aplicaciones. |                     |

Tabla 37

*Tarea de programación 8*

| Tarea                                                                                                       |                     |
|-------------------------------------------------------------------------------------------------------------|---------------------|
| Número de tarea: 8                                                                                          | Número historia: 4  |
| Nombre de tarea: Soporte de concurrencia.                                                                   |                     |
| Tipo de tarea: Desarrollo                                                                                   | Puntos estimados: 3 |
| Fecha inicio:                                                                                               | Fecha fin:          |
| Programador responsable: José Mori                                                                          |                     |
| Descripción:                                                                                                |                     |
| Se agregará la capacidad de soportar peticiones concurrentes garantizando la atomicidad de las operaciones. |                     |

**Tabla 38***Tarea de programación 9*

| Tarea                                                                                                         |                            |
|---------------------------------------------------------------------------------------------------------------|----------------------------|
| <b>Número de tarea:</b> 9                                                                                     | <b>Número historia:</b> 5  |
| <b>Nombre de tarea:</b> Registro de historial de cambios.                                                     |                            |
| <b>Tipo de tarea:</b> Desarrollo                                                                              | <b>Puntos estimados:</b> 8 |
| <b>Fecha inicio:</b>                                                                                          | <b>Fecha fin:</b>          |
| <b>Programador responsable:</b> José Mori                                                                     |                            |
| <b>Descripción:</b>                                                                                           |                            |
| Se crea una versión de los valores almacenados cada vez que los valores son actualizados a través de las APIS |                            |

**Tabla 39***Tarea de programación 10*

| Tarea                                                               |                            |
|---------------------------------------------------------------------|----------------------------|
| <b>Número de tarea:</b> 10                                          | <b>Número historia:</b> 6  |
| <b>Nombre de tarea:</b> Inicializar el servidor de configuración.   |                            |
| <b>Tipo de tarea:</b> Desarrollo                                    | <b>Puntos estimados:</b> 5 |
| <b>Fecha inicio:</b>                                                | <b>Fecha fin:</b>          |
| <b>Programador responsable:</b> José Mori                           |                            |
| <b>Descripción:</b>                                                 |                            |
| Se crea la anotación para inicializar el servidor de configuración. |                            |

**Tabla 40***Tarea de programación 11*

| Tarea                                                               |                            |
|---------------------------------------------------------------------|----------------------------|
| <b>Número de tarea:</b> 11                                          | <b>Número historia:</b> 6  |
| <b>Nombre de tarea:</b> Inicializar el cliente de la configuración. |                            |
| <b>Tipo de tarea:</b> Desarrollo                                    | <b>Puntos estimados:</b> 5 |
| <b>Fecha inicio:</b>                                                | <b>Fecha fin:</b>          |
| <b>Programador responsable:</b> José Mori                           |                            |
| <b>Descripción:</b>                                                 |                            |
| Se crea la anotación para inicializar el cliente de configuración.  |                            |

**Tabla 41***Tarea de programación 12*

| Tarea                                                                                                                                                            |                            |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------|
| <b>Número de tarea:</b> 12                                                                                                                                       | <b>Número historia:</b> 7  |
| <b>Nombre de tarea:</b> Cliente para leer la configuración expuesta por el servidor de configuración.                                                            |                            |
| <b>Tipo de tarea:</b> Desarrollo                                                                                                                                 | <b>Puntos estimados:</b> 5 |
| <b>Fecha inicio:</b>                                                                                                                                             | <b>Fecha fin:</b>          |
| <b>Programador responsable:</b> José Mori                                                                                                                        |                            |
| <b>Descripción:</b>                                                                                                                                              |                            |
| Se crea un cliente para leer la configuración del servicio REST, los valores utilizados para configurar el cliente se encuentran en un archivo properties o yml. |                            |

**Tabla 42***Tarea de programación 13*

| Tarea                                                                                                                        |                            |
|------------------------------------------------------------------------------------------------------------------------------|----------------------------|
| <b>Número de tarea:</b> 13                                                                                                   | <b>Número historia:</b> 7  |
| <b>Nombre de tarea:</b> Carga de variables en el contexto de la aplicación.                                                  |                            |
| <b>Tipo de tarea:</b> Desarrollo                                                                                             | <b>Puntos estimados:</b> 8 |
| <b>Fecha inicio:</b>                                                                                                         | <b>Fecha fin:</b>          |
| <b>Programador responsable:</b> José Mori                                                                                    |                            |
| <b>Descripción:</b>                                                                                                          |                            |
| Se carga la información de la configuración en el contexto de la aplicación para poder ser utilizada con la anotación @Value |                            |

**Tabla 43***Tarea de programación 14*

| Tarea                                                                                           |                             |
|-------------------------------------------------------------------------------------------------|-----------------------------|
| <b>Número de tarea:</b> 14                                                                      | <b>Número historia:</b> 8   |
| <b>Nombre de tarea:</b> Publicador de los cambios de la configuración.                          |                             |
| <b>Tipo de tarea:</b> Desarrollo                                                                | <b>Puntos estimados:</b> 13 |
| <b>Fecha inicio:</b>                                                                            | <b>Fecha fin:</b>           |
| <b>Programador responsable:</b> José Mori                                                       |                             |
| <b>Descripción:</b>                                                                             |                             |
| Se crea un publicador para los cambios (a través de APIs expuestas) de los valores almacenados. |                             |

Tabla 44

Tarea de programación 15

| Tarea                                                                                               |                     |
|-----------------------------------------------------------------------------------------------------|---------------------|
| Número de tarea: 15                                                                                 | Número historia: 8  |
| Nombre de tarea: Suscriptor de los cambios de la configuración.                                     |                     |
| Tipo de tarea: Desarrollo                                                                           | Puntos estimados: 8 |
| Fecha inicio:                                                                                       | Fecha fin:          |
| Programador responsable: José Mori                                                                  |                     |
| Descripción:                                                                                        |                     |
| Se crea un subcriptor para los cambios (a través de las APIs expuestas) de los valores almacenados. |                     |

Tabla 45

Tarea de programación 16

| Tarea                                                   |                      |
|---------------------------------------------------------|----------------------|
| Número de tarea: 16                                     | Número historia: 9   |
| Nombre de tarea: Soporte para múltiples tenants.        |                      |
| Tipo de tarea: Desarrollo                               | Puntos estimados: 13 |
| Fecha inicio:                                           | Fecha fin:           |
| Programador responsable: José Mori                      |                      |
| Descripción:                                            |                      |
| Se agregará la capacidad de soportar múltiples tenants. |                      |

**Tabla 46***Tarea de programación 17*

| Tarea                                                                                                                        |                            |
|------------------------------------------------------------------------------------------------------------------------------|----------------------------|
| <b>Número de tarea:</b> 17                                                                                                   | <b>Número historia:</b> 9  |
| <b>Nombre de tarea:</b> Carga de tenants en el contexto de la aplicación.                                                    |                            |
| <b>Tipo de tarea:</b> Desarrollo                                                                                             | <b>Puntos estimados:</b> 5 |
| <b>Fecha inicio:</b>                                                                                                         | <b>Fecha fin:</b>          |
| <b>Programador responsable:</b> José Mori                                                                                    |                            |
| <b>Descripción:</b>                                                                                                          |                            |
| Se carga la información de la configuración en el contexto de la aplicación para poder ser utilizada con la anotación @Value |                            |

**Tabla 47***Tarea de programación 18*

| Tarea                                                                          |                            |
|--------------------------------------------------------------------------------|----------------------------|
| <b>Número de tarea:</b> 18                                                     | <b>Número historia:</b> 10 |
| <b>Nombre de tarea:</b> Soporte para la gestión de plantillas para los tenants |                            |
| <b>Tipo de tarea:</b> Desarrollo                                               | <b>Puntos estimados:</b> 8 |
| <b>Fecha inicio:</b>                                                           | <b>Fecha fin:</b>          |
| <b>Programador responsable:</b> José Mori                                      |                            |
| <b>Descripción:</b>                                                            |                            |
| Se agregará la capacidad de crear tenants a partir de plantillas.              |                            |

Tabla 48

Tarea de programación 19

| Tarea                                                              |                     |
|--------------------------------------------------------------------|---------------------|
| Número de tarea: 19                                                | Número historia: 10 |
| Nombre de tarea: Exposición del servicio REST para las plantillas. |                     |
| Tipo de tarea: Desarrollo                                          | Puntos estimados: 5 |
| Fecha inicio:                                                      | Fecha fin:          |
| Programador responsable: José Mori                                 |                     |
| Descripción:                                                       |                     |
| Se crea una API REST para gestionar las plantillas.                |                     |

Tabla 49

Tarea de programación 20

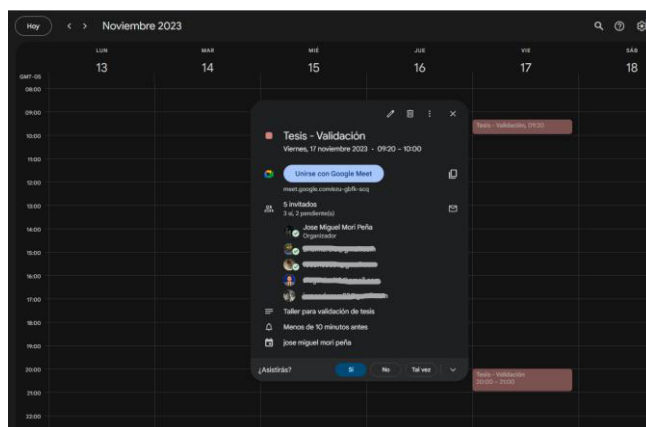
| Tarea                                               |                      |
|-----------------------------------------------------|----------------------|
| Número de tarea: 20                                 | Número historia: 11  |
| Nombre de tarea: Refactorizar APIs y Servicios.     |                      |
| Tipo de tarea: Desarrollo                           | Puntos estimados: 21 |
| Fecha inicio:                                       | Fecha fin:           |
| Programador responsable: José Mori                  |                      |
| Descripción:                                        |                      |
| Se crea una API REST para gestionar las plantillas. |                      |

## Anexo 6 Capturas del taller de validación del framework

La invitación de los participantes se realizó vía telefónica y agendada a través de Google Meet en dos reuniones como se observa en la Figura 79 y en la Figura 80.

**Figura 79**

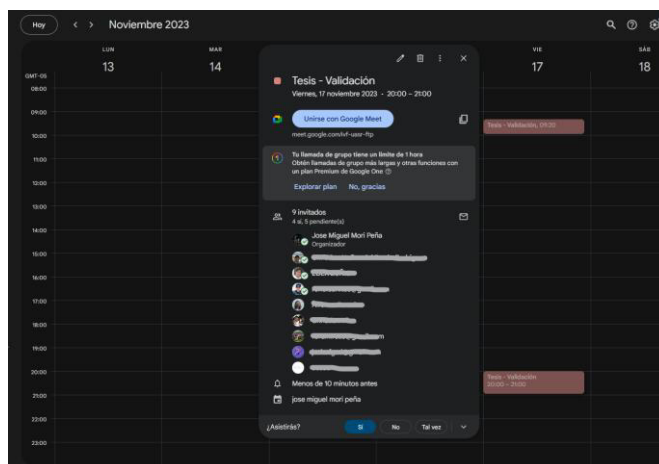
*Evento - primer grupo*



*Nota.* Evento de Google Meet realizado el 17 de noviembre del 2023 con el primer grupo de participantes.

**Figura 80**

*Evento - segundo grupo*



*Nota.* Evento de Google Meet realizado el 17 de noviembre del 2023 con el segundo grupo de participantes.

Capturas de pantalla de la presentación del proyecto, framework y aplicación de este al primer grupo se muestran en la Figura 81, Figura 82, Figura 83, Figura 84 y Figura 85.

**Figura 81**

*Presentación - primer grupo I*

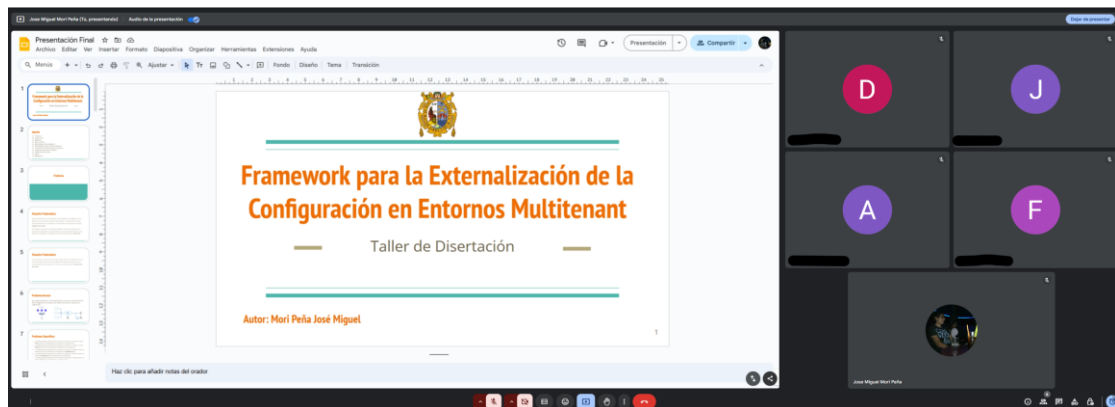
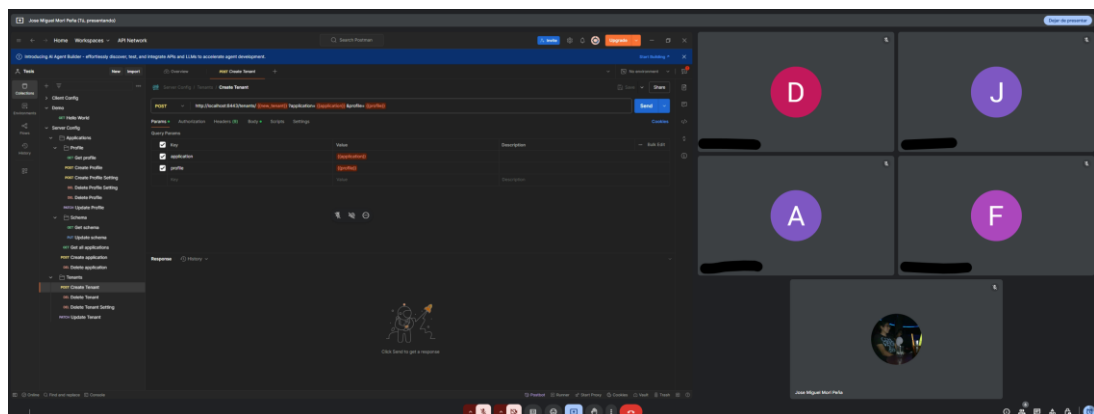
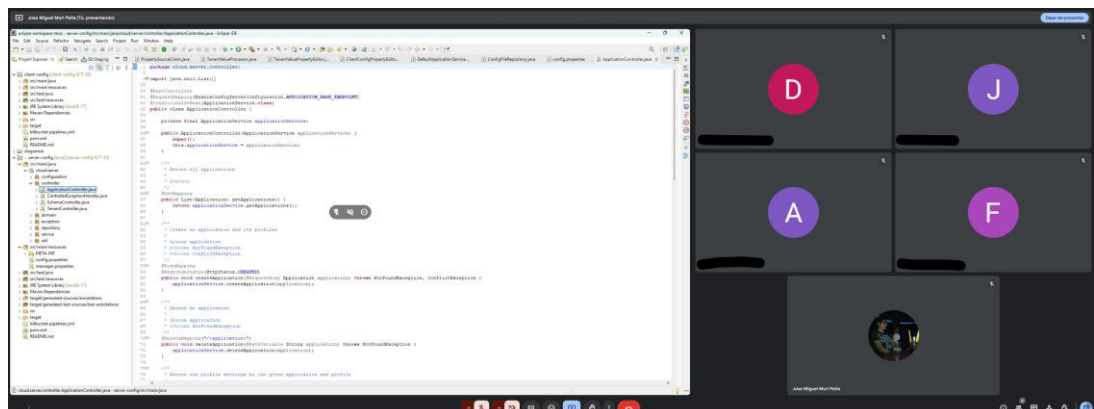


**Figura 82**

*Presentación - primer grupo II*



**Figura 83***Presentación - primer grupo III***Figura 84***Presentación - primer grupo IV***Figura 85***Presentación - primer grupo V*

Capturas de pantalla de la presentación del proyecto, framework y aplicación de este al segundo grupo se muestran en la Figura 86, Figura 87, Figura 88 y Figura 89

**Figura 86**

*Presentación - segundo grupo I*

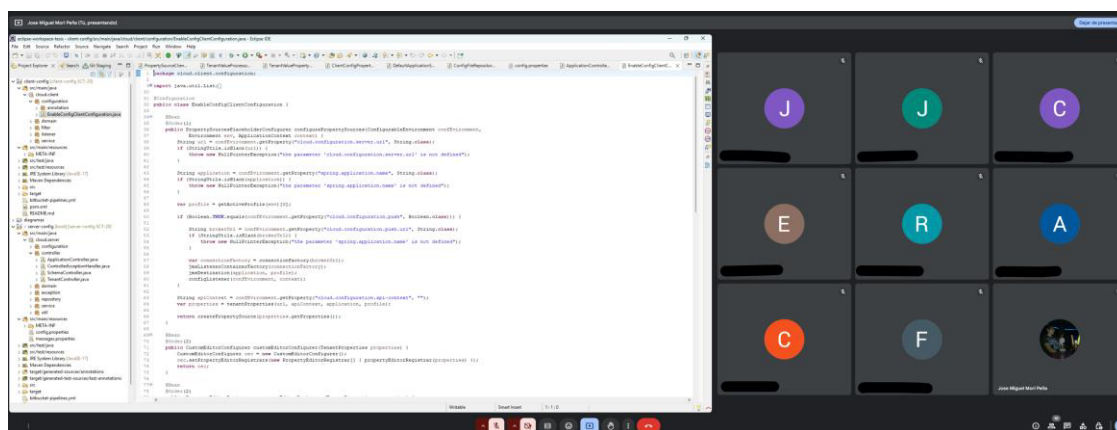


**Figura 87**

*Presentación - segundo grupo II*

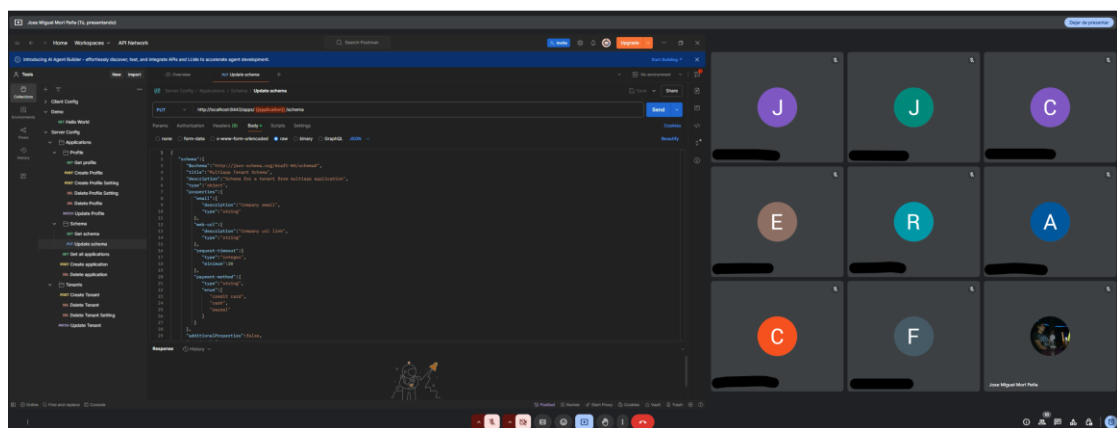


**Figura 88**

*Presentación - segundo grupo III*

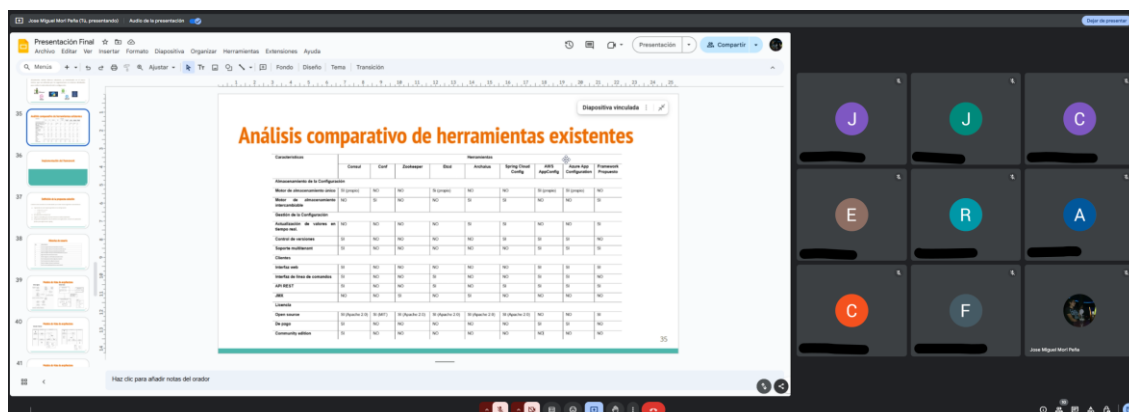
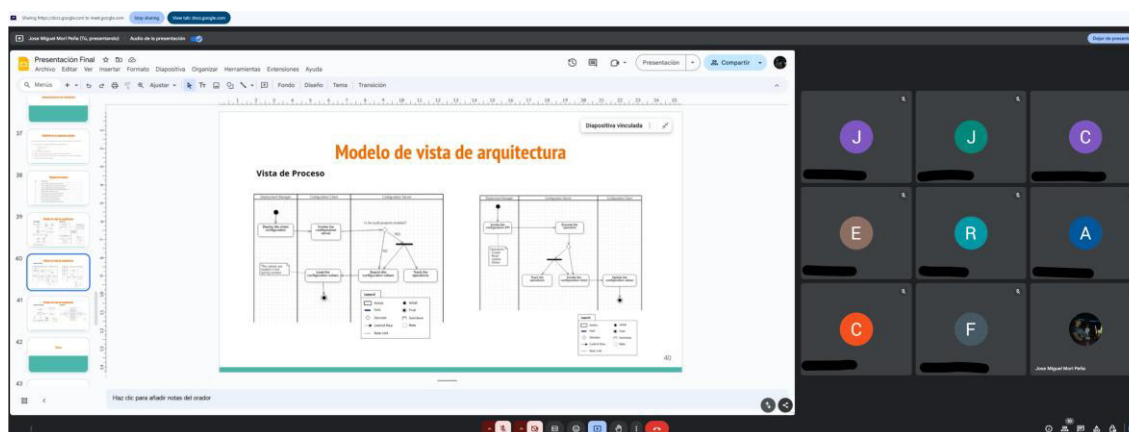


**Figura 89**

*Presentación - segundo grupo IV*



## Anexo 7 Proyecto para el taller practico

Se crea un proyecto maven multi modulo, el cual incluye los módulos micro-server y micro-client, los cuales sirven como proyecto cliente y servidor respectivamente, dichos módulos son proyectos web básicos que son ejecutados en el servidor tomcat.

En la Figura 90 se muestran las dependencias de ambos proyectos web con spring boot.

**Figura 90**

*Dependencias del proyecto multi modulo*

```
<dependencyManagement>
  <dependencies>
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-dependencies</artifactId>
      <version>3.3.3</version>
      <type>pom</type>
      <scope>import</scope>
    </dependency>
  </dependencies>
</dependencyManagement>

<modules>
  <module>micro-server</module>
  <module>micro-client</module>
</modules>

<dependencies>
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-web</artifactId>
  </dependency>

  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-data-jpa</artifactId>
  </dependency>

  <dependency>
    <groupId>com.h2database</groupId>
    <artifactId>h2</artifactId>
    <scope>runtime</scope>
  </dependency>

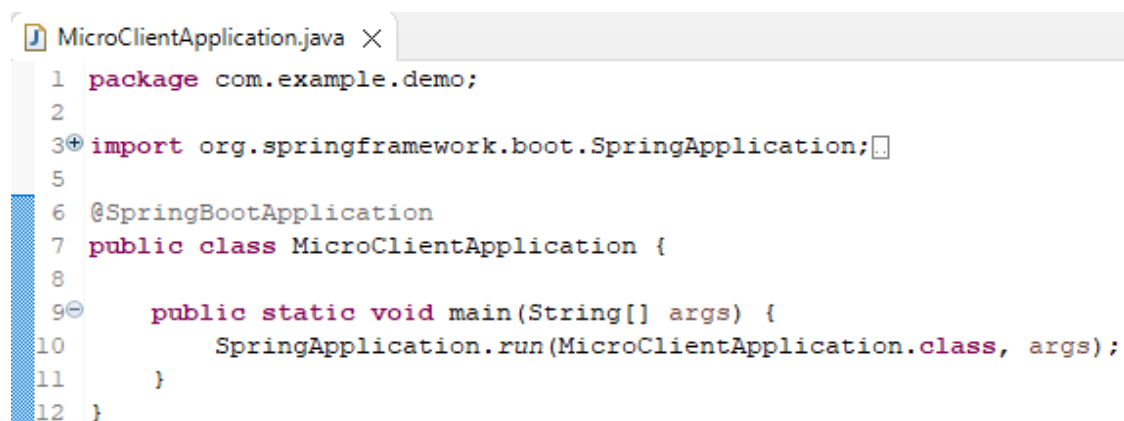
  <dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-test</artifactId>
    <scope>test</scope>
  </dependency>
</dependencies>
```

*Notas.* Parte del archivo pom.xml del proyecto multi modulo que incluye las dependencias básicas para ejecutar un proyecto web.

Cada módulo solo cuenta con su clase principal para su funcionamiento como se observa en la Figura 91 y Figura 92.

### Figura 91

#### *Clase principal del módulo micro client*

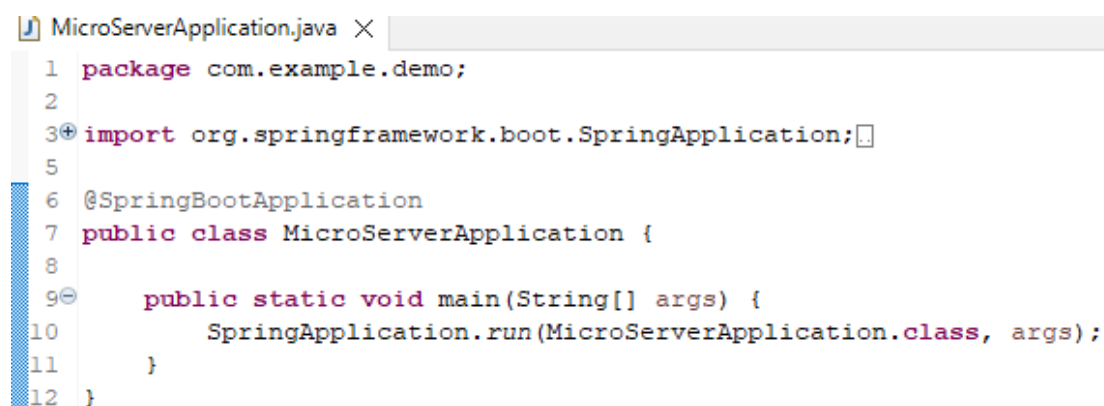


```
1 package com.example.demo;
2
3 import org.springframework.boot.SpringApplication;
4
5
6 @SpringBootApplication
7 public class MicroClientApplication {
8
9     public static void main(String[] args) {
10         SpringApplication.run(MicroClientApplication.class, args);
11     }
12 }
```

*Nota.* La clase solo necesita la anotación y la llamada al método estático para su funcionamiento.

### Figura 92

#### *Clase principal del módulo micro server*



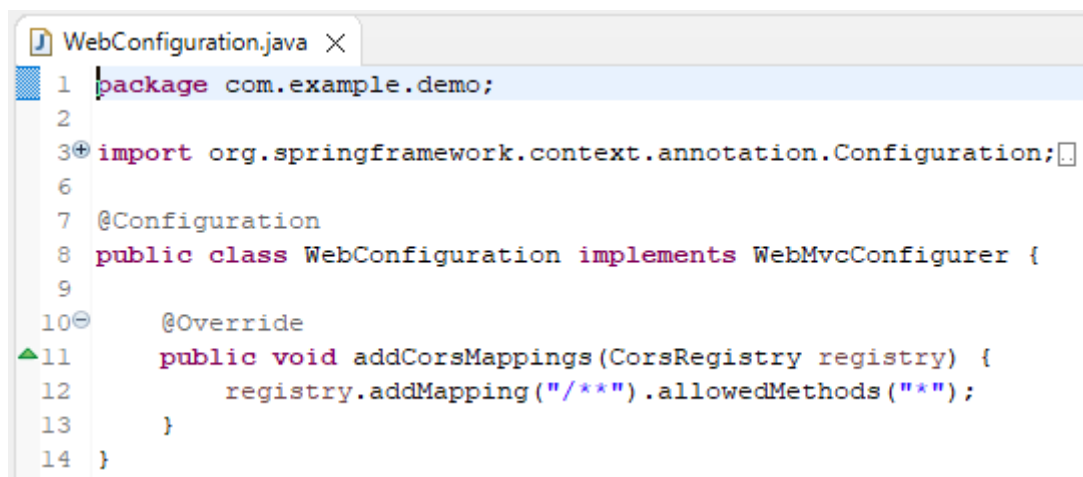
```
1 package com.example.demo;
2
3 import org.springframework.boot.SpringApplication;
4
5
6 @SpringBootApplication
7 public class MicroServerApplication {
8
9     public static void main(String[] args) {
10         SpringApplication.run(MicroServerApplication.class, args);
11     }
12 }
```

*Nota.* La clase solo necesita la anotación y la llamada al método estático para su funcionamiento.

Adicionalmente, se agrega la clase de la Figura 93 en el módulo micro server para especificar los orígenes permitidos, encabezados y métodos permitidos.

**Figura 93**

*Clase para configurar cors*



```
WebConfiguration.java X
1 package com.example.demo;
2
3 import org.springframework.context.annotation.Configuration;
4
5
6
7 @Configuration
8 public class WebConfiguration implements WebMvcConfigurer {
9
10     @Override
11     public void addCorsMappings(CorsRegistry registry) {
12         registry.addMapping("/**").allowedMethods("*");
13     }
14 }
```

Esta clase permite que la aplicación hecha en angular pueda invocar a las APIs del módulo micro server una vez exponga las APIs proveídas por el framework.

## Anexo 8 Materiales de capacitación

### 1. Guía de uso del config manager (server)

El archivo README.md provee la información para instalar y configurar el framework en el proyecto servidor, se detalla su contenido en la Figura 94, y adicionalmente se brinda más detalles sobre personalización avanzada en la Figura 95.

#### Figura 94

##### *Archivo readme.md del config manager - server I*

The framework gives you the ability to externalize your project's configuration, providing three (files, jdbc and redis) destinations to store it.

#### Installation

Add a maven dependency in the `pom.xml` of your project:

```
<dependency>
  <groupId>cloud</groupId>
  <artifactId>server-config</artifactId>
  <version>0.0.1-SNAPSHOT</version>
</dependency>
```

#### Configuration

- Add the below properties in the `application.yml` of your project:

```
# Optional. enable the flow to update the configuration in real time.
cloud.configuration.push=true

# Mandatory when the above property is TRUE. Set the the queue url that will be used to interchange message between the ser
cloud.configuration.push.url=tcp://localhost:61616

# Optional. enable the detailed logs.
cloud.configuration.log=true

# Mandatory, set the storage source type (file, jdbc or redis)
cloud.configuration.source=file

# Mandatory when the source is file, Set the directory path to store the files that will contain the configuration.
cloud.configuration.source.file.path=/home/josem/eclipse-workspace-tesis/source/

# Optional, Set a prefix to the given APIs exposed by the server.
cloud.configuration.api-context=
```

**Note** The source `jdbc` and `redis` need a `javax.sql.DataSource` and `redis.clients.jedis.JedisPool` instance respectively to get connection with the selected engine.

- \* Add the below annotation in your main class

```
@EnableConfigurationServer
```

## Figura 95

### *Archivo readme.md del config manager - server II*

#### Advance Customization

If you use `jdb` to store the configuration, you can overwrite any of the below properties to customize the table and columns where the configuration will be stored in your transaction database.

```
cloud.configuration.jdbc.application.table-name=cloud_config

cloud.configuration.jdbc.application.sql-select-by-application-profile=
cloud.configuration.jdbc.application.sql-select-by-application-profile-tenant=
cloud.configuration.jdbc.application.sql-delete-by-application-profile=
cloud.configuration.jdbc.application.sql-delete-by-application-profile=

cloud.configuration.jdbc.application.sql-select-by-application-profile-key=
cloud.configuration.jdbc.application.sql-delete-by-application-profile-key=
cloud.configuration.jdbc.application.sql-update-by-application-profile-key=

cloud.configuration.jdbc.application.sql-delete-by-application-profile-tenant=

cloud.configuration.jdbc.application.sql-delete-by-application-profile-tenant-key=
cloud.configuration.jdbc.application.sql-update-by-application-profile-tenant-key=
cloud.configuration.jdbc.application.sql-select-by-application-profile-tenant-key=

cloud.configuration.jdbc.application.sql-insert=
cloud.configuration.jdbc.application.sql-select-applications=
cloud.configuration.jdbc.application.sql-delete-by-application=

#TEMPLATE
cloud.configuration.jdbc.schema.table-name=
cloud.configuration.jdbc.schema.sql-select-by-application=
cloud.configuration.jdbc.schema.sql-insert=
cloud.configuration.jdbc.schema.sql-updateByApplication=
```

## 2. Guía de uso del config manager (client)

En la Figura 96 se muestra el archivo README.md del proyecto cliente donde se detalla su instalación y configuración.

**Figura 96**

### *Archivo readme.md del config manager - client*

The framework gives your project the ability to consume the configuration provided by the `Config Manager - Server`.

#### Installation

Add a maven dependency in the `pom.xml` of your project:

```
<dependency>
  <groupId>cloud</groupId>
  <artifactId>client-config</artifactId>
  <version>0.0.1-SNAPSHOT</version>
</dependency>
```

#### Configuration

- Add the below properties in the `application.yml` of your project:

```
# Optional. enable the flow to update the configuration in real time.
cloud.configuration.push=true

# Mandatory when the above property is TRUE. Set the the queue url that will be used to interchange message between the ser
cloud.configuration.push.url=tcp://localhost:61616

# Mandatory. set the url of the configuration server that will provide the configuration.
cloud.configuration.server.url=http://localhost:8443

# Optional, Set a prefix to the given APIs exposed by the server, this must match with the value defined in the server proj
cloud.configuration.api-context=
```

\* Add the below annotation in your main class

```
@EnableConfigurationClient
```

### 3. Colección de APIs en postman

La colección de APIs para gestionar la configuración de las aplicaciones es compartida a través de la herramienta Postman en formato json como se observa en la Figura 97.

**Figura 97**

*Colección de APIs - postman*

