Pontificia Universidad Javeriana Cali
Facultad de Ingeniería.
Maestría en Ingeniería de Software
Proyecto de Grado.

Diseño de una Arquitectura Integrada para un Sistema de Agendamiento Multi-Sala de Videoconferencias en Campus Universitarios: Implementación en Zoom y Teams.

Alexander López Téllez

Director: Mg. Jhon Henry Gómez Tabares

27 de Julio de 2025

Santiago de Cali, 27 de Julio de 2025.

Señores
Pontificia Universidad Javeriana Cali.
Ph.D. Luisa Rincón
Directora Maestría en Ingeniería de Software.
Cali.

Cordial Saludo.

Por medio de la presente hago constar que en mi calidad de director de trabajo de grado he revisado el proyecto titulado "Diseño de una Arquitectura Integrada para un Sistema de Agendamiento Multi-Sala de Videoconferencias en Campus Universitarios: Implementación en Zoom y Teams." realizado por el estudiante de Magister en Ingeniería de Software Alexander López Téllez (cod: 8990232), el cual se encuentra terminado y considero que cumple con los requisitos para ser sustentado.
Atentamente,

Mg. Jhon Henry Gómez Tabares

Santiago de Cali, 27 de Julio de 2025.

Señores
Pontificia Universidad Javeriana Cali
Ph.D. Luisa Rincón
Directora Maestría en Ingeniería de Software
Cali.

Cordial Saludo.

Me permito presentar a su consideración el proyecto de grado titulado "Diseño de una Arquitectura Integrada para un Sistema de Agendamiento Multi-Sala de Videoconferencias en Campus Universitarios: Implementación en Zoom y Teams." con el fin de cumplir con los requisitos exigidos por la Universidad y para que sea sometido a revisión del jurado y cumpla su aprobación, para conseguir posteriormente el título de Magister en Ingeniería de Software.

Atentamente,

Alexander López Téllez
Código: 8990232

Ficha Resumen
Trabajo de Grado Maestría en Ingeniería de Software

TíTULO: Diseño de una Arquitectura Integrada para un Sistema de Agendamiento Multi-Sala de Videoconferencias en Campus Universitarios: Implementación en Zoom y Teams.

1. Énfasis: Ingeniería de Software
2. Área de trabajo: Ingeniería
3. Tipo de proyecto (Aplicado, Innovación, Investigación):
4. Estudiante: Alexander López Téllez
5. Correo electrónico: alexander@javerianacali.edu.co
6. Dirección y teléfono: Carrera 1 B1 # 52 A Bis 117 - 3202316574
7. Director: Mg. Jhon Henry Gómez Tabares
8. Vinculación del director: (Cátedra)
9. Correo electrónico del director: Jhon.gomezt@javerianacali.edu.co
10. Co-Director (Si aplica): No
11. Grupo o empresa que lo avala (Si aplica): No
12. Otros grupos o empresas: No
13. Palabras clave(al menos 5): Videoconferencing, automatic scheduling, higher education, system integration, cybersecurity.
14. ODS que aplica al proyecto (Agenda 2030): ODS 4 - Educación de Calidad.
15. Fecha de inicio: 2/07/2024
16. Resumen: Esta tesis desarrolla un sistema de agendamiento de videoconferencias diseñado para integrarse con plataformas como Zoom y Microsoft Teams, específicamente adaptado a las necesidades de la educación superior. Este sistema se enfoca en superar las insuficiencias de las soluciones actuales, especialmente en términos de personalización, capacidad y seguridad, que son esenciales para los campus universitarios en contextos de alta demanda. El diseño propuesto es escalable, permitiendo la gestión automática de la creación de muúltiples salas de videoconferencia cuando se supera la capacidad máxima permitida por sala, garantizando

así una integración efectiva con los sistemas de información y gestión académica de las universidades. Esta solución se destaca por su capacidad para adaptarse a las especificidades y la identidad operativa de cada institución educativa.

El documento de tesis detalla el proceso completo desde el análisis de las limitaciones actuales de las plataformas de videoconferencia hasta el desarrollo del diseño arquitectónico del sistema. Se realiza una revisión de la literatura y se aplican metodologías de investigación mixtas para recopilar y analizar datos que fundamenten el diseño. El modelo C4 se utilizará para desarrollar un diseño conceptual detallado en los niveles de contexto, contenedores y componentes, lo que permite documentar con precisión la arquitectura propuesta y preparar el terreno para futuras implementaciones que podrían extenderse más allá del ámbito académico hacia aplicaciones corporativas y gubernamentales.

Agradecimientos

En primer lugar, agradezco a Dios, mi guía y sustento, por concederme una segunda oportunidad de vida. Gracias por darme las fuerzas necesarias cuando el cuerpo flaqueaba, por el discernimiento en los momentos de confusión, por la sabiduría que me permitió avanzar con firmeza, pero, sobre todo, por la berraquera que sembraste en mi corazón para continuar con determinación este camino académico, desde el pregrado hasta culminar esta etapa de posgrado. Su presencia ha sido constante, viva y real en cada paso, y a Él le atribuyo toda gloria por este logro.

A mi esposa, compañera fiel y apoyo incondicional, le agradezco desde lo más profundo de mi ser. Su sacrificio silencioso, asumiendo labores adicionales en el hogar y en la familia, fue el fundamento que me permitió disponer de tiempo para estudiar. Su amor, paciencia y entrega fueron faro constante en las noches de cansancio y desvelo. Sin su compañía, este trayecto no habría sido posible.

A mis hijos, quienes me acompañaron durante largas jornadas de estudio, regalándome besos y abrazos que se convirtieron en la energía que necesitaba para continuar. Gracias por comprender mis ausencias, por permitirme ser ejemplo en sus vidas y por enseñarme, sin saberlo, a ser mejor padre, mejor persona y mejor estudiante. Ellos son, sin duda, una de mis más grandes motivaciones para seguir creciendo, no solo a nivel personal y familiar, sino también académico.

A mi madre, mi primera maestra de vida, que desde la distancia me ha sostenido con su amor incondicional. Fue ella quien me levantó cuando todo parecía perdido, quien creyó en mí incluso cuando yo dudaba, y quien me inculcó desde niño la importancia del esfuerzo, la humildad y la fe. Hoy soy quien soy gracias a su valentía, entrega y palabra constante de aliento.

A mi abuelita Isabel, que habita eternamente en mi corazón. Aunque físicamente ya no me acompaña, su voz resuena en mi interior en los momentos de tristeza y desaliento. Su legado espiritual y la palabra que dejó grabada en mí —"Mira que te mando que te esfuerces y seas valiente..."(Josué 1:9)— han sido guía firme en los valles de sombra y duda.

Extiendo un agradecimiento especial a mi asesor de tesis, el Ingeniero Jhon H. Gómez T., por su acompañamiento cercano, sus observaciones acertadas, su guía profesional y su compromiso con el proceso. Su orientación ha sido clave para la construcción rigurosa y sólida de este trabajo de investigación.

A mi compañero de maestría Edgar Roa, quien, más allá de ser colega, se convirtió en un amigo en este camino. Su generosidad al compartir su conocimiento, su disposición desinteresada para explicarme temas complejos del desarrollo de software, y sus palabras de ánimo en momentos críticos fueron un apoyo invaluable. Gracias por tu fraternidad académica y personal.

Finalmente, agradezco profundamente a todo el cuerpo docente de la Maestría en Ingeniería de Software. De cada uno he aprendido no solo contenidos técnicos, sino también estrategias de comunicación, pedagogía y estructura académica. Me llevo estos aprendizajes no solo como profesional, sino también como docente y ahora como maestro, con la firme intención de compartirlos con cada

estudiante a quien tenga el privilegio de servir.

Gracias a todos los que, de una u otra forma, hicieron parte de este proceso. Este logro es también suyo.

Resumen

Esta tesis propone el desarrollo de un sistema de agendamiento de videoconferencias diseñado para mejorar la integración de plataformas como Zoom y Microsoft Teams con los sistemas de autenticación de los campus universitarios. Destaca la necesidad de un sistema capaz de gestionar de manera eficiente múltiples salas de videoconferencia simultáneas, garantizando una autenticación segura de los usuarios, y abordando las limitaciones comunes de capacidad y personalización. El objetivo principal es diseñar un marco arquitectónico que automatice la programación de sesiones y garantice una integración segura, mejorando así la experiencia educativa en línea y satisfaciendo las necesidades operativas de las universidades. Se anticipa que los resultados incluirán una arquitectura escalable capaz de manejar altos volúmenes de tráfico de datos y múltiples usuarios simultáneos, manteniendo un rendimiento sólido y la satisfacción del usuario. Esta investigación sienta las bases para futuros avances en herramientas de comunicación digital, con aplicaciones potenciales que se extienden más allá del ámbito educativo, incluyendo entornos corporativos y gubernamentales.

Palabras Clave: Videoconferencias, programación automática, educación superior, integración de sistemas, ciberseguridad.

Abstract

This thesis outlines the development of a videoconferencing scheduling system designed to enhance the integration of platforms such as Zoom and Microsoft Teams with university campus authentication systems. It addresses the necessity for a system capable of managing multiple simultaneous videoconferencing rooms efficiently and ensuring secure user authentication, thereby overcoming existing limitations related to capacity and customization. The primary aim is to engineer an architectural framework that supports automatic session scheduling and robust integration, thus enhancing the online educational experience and meeting the specific operational requirements of each university. The anticipated outcomes are a scalable architecture adept at handling substantial data traffic and numerous concurrent users, while maintaining high performance and user satisfaction. This research paves the way for broader applications, laying a foundation for the advancement of digital communication tools across educational, corporate, and governmental sectors.
Keywords: Videoconferencing, customization, scalability, higher education, digital communication, automatic scheduling.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

Las plataformas de videoconferencia como Zoom y Microsoft Teams se han consolidado como herramientas esenciales para la educación en línea, especialmente tras la pandemia de COVID-19. Si bien estas plataformas han permitido la continuidad académica a nivel global, diversos estudios, como los de Lowenthal et al. (2020), García-Morales et al. (2021) y Riyath & Rajapakse. (2022), han identificado limitaciones críticas en términos de personalización, escalabilidad y capacidad de gestión para entornos educativos con grandes volúmenes de usuarios. Un ejemplo de ello es el contexto de esta investigación, donde un curso universitario con aproximadamente 27,000 estudiantes enfrenta restricciones debido a licencias que solo permiten hasta 10,000 conexiones simultáneas, evidenciando un desafío arquitectónico en la administración de recursos y sesiones virtuales.

En este contexto, la presente tesis propone el diseño de una arquitectura tecnológica para un sistema de agendamiento de videoconferencias. El enfoque arquitectónico busca establecer los lineamientos estructurales para un sistema que optimice la integración con los sistemas de gestión académica y autenticación de los campus universitarios. En lugar de desarrollar una aplicación específica, la investigación se centra en la definición, modelado y documentación de una infraestructura tecnológica, proporcionando una base conceptual para futuras implementaciones. A diferencia de una solución de software tradicional, este trabajo se centra en la concepción y modelado de una infraestructura tecnológica que optimice la gestión de sesiones virtuales en función de criterios de escalabilidad, seguridad y eficiencia, permitiendo una mejor asignación de recursos en entornos de alta demanda.

Los objetivos de esta investigación incluyen, en primer lugar, la identificación de las principales limitaciones de las plataformas de videoconferencia en términos de capacidad y gestión de múltiples usuarios, mediante un enfoque mixto que combine revisión de literatura y recolección de datos empíricos. Posteriormente, se definirán los atributos de calidad y requisitos arquitectónicos para el diseño del sistema de agendamiento, estableciendo un marco de referencia basado en estándares de arquitectura de software y mejores prácticas en la industria. En esta investigación, se desarrollará un diseño arquitectónico para un sistema de agendamiento de videoconferencias que optimice la integración con plataformas como Zoom y Microsoft Teams, garantizando escalabilidad, seguridad y eficiencia en entornos educativos de gran demanda. Una vez establecida la arquitectura del sistema, se llevará a cabo un análisis comparativo entre Amazon AWS y Microsoft Azure para determinar cuál de estas plataformas proporciona los servicios más adecuados para su implementación. Este análisis estará alineado con los atributos de calidad y requisitos definidos previamente en el diseño arquitectónico, asegurando que la nube seleccionada sea la mejor opción para soportar la

infraestructura conceptualizada. Finalmente, se documentará el diseño arquitectónico y se definirán lineamientos sobre cómo podría ser implementado en la plataforma seleccionada.

En cuanto a la metodología, esta investigación empleará un enfoque mixto. Se realizarán entrevistas y encuestas a usuarios clave (docentes, administradores y estudiantes) para recopilar información sobre sus experiencias y necesidades en el uso de videoconferencias a gran escala. Además, se efectuará un análisis comparativo de las principales plataformas en la nube, considerando sus características técnicas y operativas en relación con los requerimientos del sistema de agendamiento. Como resultado, se generará un conjunto de modelos arquitectónicos, utilizando técnicas de diseño como el modelo C4 para representar los diferentes niveles de abstracción del sistema.

La relevancia de esta investigación radica en su contribución al diseño de arquitecturas escalables y eficientes para la gestión de videoconferencias en entornos educativos. Al optimizar la administración de sesiones y mejorar la experiencia de usuario, este proyecto no solo beneficiará a las instituciones académicas, sino que también podrá extenderse a sectores corporativos y gubernamentales con necesidades similares.

Finalmente, la tesis se estructurará en los siguientes capítulos:

1. **Capítulo 1: Introducción**. Presenta el contexto del problema, la justificación y los objetivos de la investigación.
2. **Capítulo 2: Marco de referencia**. Analiza estudios previos relacionados con las plataformas de videoconferencia y sus limitaciones en entornos educativos.
3. **Capítulo 3: Desarrollo del proyecto**. Describe el enfoque metodológico utilizado, así como los métodos de recolección y análisis de datos.
4. **Capítulo 4: Evaluación**. Expone el proceso de diseño y modelado de la arquitectura del sistema propuesto.
5. **Capítulo 5: Conclusiones**. Presenta los hallazgos obtenidos a partir de la evaluación del diseño arquitectónico propuesto.
6. **Capítulo 6: Cronograma**. Resume los principales aportes de la investigación y sugiere futuras líneas de trabajo.

1.1. Definición del problema

Las plataformas de videoconferencia como Zoom y Microsoft Teams han sido fundamentales para facilitar la educación en línea durante la pandemia de COVID-19, permitiendo a millones de estudiantes acceder a clases y actividades académicas a distancia. Sin embargo, a medida que las instituciones educativas, especialmente las de gran tamaño, han integrado estas plataformas a sus operaciones diarias, se han identificado limitaciones críticas que dificultan su efectividad en contextos educativos más complejos. Estas plataformas no están diseñadas específicamente para los entornos educativos amplios y diversos, lo que impide a los docentes personalizar la experiencia de enseñanza según las necesidades específicas de los estudiantes y los campus universitarios (García-Morales et al., 2021).

En particular, se ha evidenciado que la capacidad limitada de participantes por videoconferencia representa un obstáculo significativo para la escalabilidad de los cursos. Instituciones que manejan un número considerable de estudiantes enfrentan restricciones al intentar garantizar la participación de todos en las actividades académicas virtuales. Por ejemplo, un curso en el que es docente el autor, que involucra a más de 27,000 estudiantes, que actualmente utiliza licencias con un límite de 10,000 conexiones simultáneas, se ve afectado por estas limitaciones (Riyath & Rajapakse, 2022).

La situación resulta indeseable porque compromete la calidad de la enseñanza al impedir la participación de toda la matrícula. Además, genera desafíos operativos que reducen la eficacia y la accesibilidad de la educación en línea. Aunque se han ensayado soluciones parciales, aún no existe una propuesta que afronte de forma integral las limitaciones de capacidad y personalización.

Es imperativo, entonces, explorar nuevas soluciones tecnológicas que permitan una integración más efectiva con los sistemas de gestión de aprendizaje y autenticación de los campus universitarios. De esta manera, se busca desarrollar un sistema que no solo supere las limitaciones actuales de las plataformas de videoconferencia, sino que también permita a los docentes gestionar videoconferencias masivas con un enfoque en la escalabilidad, seguridad y eficiencia. La falta de soluciones adecuadas, en este sentido, sigue siendo un problema que debe resolverse para garantizar la sostenibilidad y calidad de la educación en línea a largo plazo.

1.1.1. Planteamiento del problema

Las plataformas de videoconferencia Zoom y Microsoft Teams entre las más utilizadas, se consolidaron como herramientas esenciales para la educación superior, especialmente durante la pandemia de COVID-19. Sin embargo, diversos estudios han señalado que estas soluciones aún presentan barreras que afectan la calidad de la experiencia académica en contextos de alta demanda. Por un lado, la falta de personalización y adaptación a las dinámicas específicas de cada campus limita la capacidad de los docentes para crear entornos de aprendizaje que se ajusten a las necesidades individuales de los estudiantes (Cavus & Sekyere-Asiedu, 2021). Por otro, se mantienen restricciones técnicas como topes de participantes por sesión o licencias institucionales insuficientes que resultan críticas en asignaturas con grupos masivos, donde las conexiones concurrentes superan con facilidad la capacidad contratada.

Si estas limitaciones no se abordan, la educación remota corre el riesgo de reproducir inequidades, ya que obstaculiza la interacción fluida entre estudiantes y docentes y se compromete el acceso simultáneo de toda la matrícula.

Para cuantificar la magnitud de estas limitaciones y priorizar los atributos arquitectónicos, se aplicó en diciembre de 2024 una encuesta a 31 actores clave (docentes, personal de soporte y algunos estudiantes). El instrumento (descrito en la Sección 1.5) recoge frecuencia de uso, dificultades concretas (capacidad, flexibilidad, seguridad) y percepción de la eficacia de Zoom/Teams. Los hallazgos empíricos obtenidos servirán de evidencia para la definición y la priorización de los requisitos funcionales y no funcionales del sistema propuesto.

Por ello se propone desarrollar un sistema de agendamiento de videoconferencias que, de forma automática y escalable, orqueste la creación de múltiples salas espejo e integre los mecanismos de autenticación y los sistemas de gestión académica de la universidad. Este sistema, además, deberá garantizar la privacidad y la protección de los datos personales conforme a la Ley 1581 de 2012 en Colombia.

Un componente clave de esta propuesta es la selección de la infraestructura en la nube que hospedará el sistema. Por ello, se realizará un análisis comparativo entre Amazon Web Services (AWS) y Microsoft Azure para determinar cuál plataforma ofrece la mejor combinación de escalabilidad, seguridad, rendimiento y costo total de propiedad, de modo que soporte tanto la carga actual como el crecimiento futuro de usuarios.

Al atender estas necesidades, la solución contribuirá a fortalecer la calidad e inclusión de la educación virtual en línea con el Objetivo de Desarrollo Sostenible 4 y ofrecerá un marco sostenible para instituciones que requieren gestionar clases con miles de estudiantes conectados simultáneamente, garantizando una experiencia educativa intuitiva y segura para toda la comunidad universitaria.

1.2. Objetivos del proyecto

1.2.1. Objetivo General

Desarrollar una arquitectura para un sistema de agendamiento de videoconferencias que se integre eficazmente con las plataformas de videoconferencia existentes como Zoom y Teams, y los sistemas de autenticación del campus universitario.

1.2.2. Objetivos específicos

- Identificar las limitaciones actuales en capacidad y gestión de múltiples usuarios en plataformas de videoconferencia como Zoom y Teams, realizando un análisis mixto (cualitativo y cuantitativo) que incluya revisión de literatura y recopilación de feedback de usuarios y expertos en tecnología educativa.

- Determinar los atributos de calidad necesarios para el diseño arquitectónico del sistema de agendamiento de videoconferencias, análizando su impacto en la eficiencia, seguridad y escalabilidad del sistema.

- Analizar Amazon AWS y Microsoft Azure para determinar cuál de estas plataformas proporciona los servicios más adecuados para la implementación de la arquitectura del sistema de agendamiento de videoconferencias, en función de los atributos de calidad definidos en el diseño arquitectónico previo.

- Desarrollar el diseño arquitectónico del sistema de agendamiento de videoconferencias, enfocado en la integración eficaz con plataformas de videoconferencia existentes y sistemas de autenticación del campus.

1.3. Delimitaciones y alcances

1.3.1. Análisis de Limitaciones Actuales

1. Se realizó una revisión exhaustiva de la literatura para identificar y documentar las limitaciones actuales de las plataformas de videoconferencia como Zoom y Microsoft Teams, especialmente en términos de capacidad y personalización en contextos educativos grandes. Este análisis incluyó una revisión de al menos 30 fuentes académicas relevantes, centrándose en las principales limitaciones encontradas y su impacto en entornos educativos.

2. Se aplicaron metodologías de investigación mixtas, que incluyeron encuestas y entrevistas con usuarios finales y expertos en educación virtual, para recoger y analizar datos sobre las limitaciones percibidas y las necesidades específicas de los usuarios. Se realizaron entrevistas con al menos 30 usuarios finales y expertos en educación virtual, y se presentó un informe con los hallazgos clave y sugerencias para abordar las limitaciones.

3. Para este proyecto, se diseñaron e implementaron protocolos que garantizan el cumplimiento del marco legal colombiano para la protección de datos personales, establecidos por la legislación colombiana. Esto incluye:

   - Asegurarse de que todos los usuarios proporcionen su consentimiento explícito antes de que sus datos personales sean procesados.

   - Implementación de medidas tecnológicas para proteger la información personal de accesos no autorizados, alteraciones o pérdidas.

   - Implementación de mecanismos para que los usuarios accedan a sus datos personales y soliciten su corrección o supresión cuando sea necesario.

4. Se integraron los resultados obtenidos para formar una base sólida que guió el desarrollo de la arquitectura del sistema de agendamiento. Se desarrolló un marco conceptual basado en los resultados que apoyó el diseño y la implementación del sistema de agendamiento.

1.3.2. Desarrollo de Diseño Conceptual

1. Se utilizó la herramienta de modelado C4 para desarrollar un diseño conceptual detallado del sistema de agendamiento de videoconferencias en los niveles de contexto, contenedores y componentes. Se elaboraron diagramas de los tres niveles mencionados, que demostraron claramente la estructura y funcionamiento del sistema propuesto.

2. Se elaboraron documentos que incluyeron la definición de las interacciones del sistema con los usuarios y otros sistemas en el Nivel de Contexto, la descripción de las aplicaciones y servicios de alto nivel que componen el sistema en el Nivel de Contenedores, y la explicación de los

componentes internos de estos contenedores en el Nivel de Componentes. Se produjo documentación completa que incluyó diagramas, descripciones técnicas detalladas y justificaciones de las decisiones de diseño.

3. Una vez desarrollado el diseño arquitectónico del sistema de agendamiento de videoconferencias, se llevó a cabo un análisis comparativo entre Amazon AWS y Microsoft Azure con el objetivo de seleccionar la plataforma más adecuada para su implementación. Esta evaluación se realizó en función de los atributos técnicos y económicos definidos en el diseño arquitectónico, incluyendo escalabilidad, seguridad, costos operativos y compatibilidad con infraestructuras de gestión de aprendizaje existentes.

1.3.3. Supuestos del Proyecto

En el desarrollo de este proyecto de agendamiento de videoconferencias para entornos educativos, se consideraron los siguientes supuestos que fundamentan el marco de trabajo y las expectativas del proyecto:

1. Se asumió que todos los recursos tecnológicos y humanos necesarios, incluyendo acceso a plataformas de videoconferencia como Zoom y Microsoft Teams, estarían disponibles durante el periodo de desarrollo del proyecto.

2. Se esperó una colaboración activa de los usuarios finales y expertos en educación virtual para la recopilación de datos y pruebas de la arquitectura propuesta. Esto incluye la disposición de las partes interesadas para participar en entrevistas, encuestas y sesiones de validación.

3. Se supuso que no habría cambios significativos en las plataformas de videoconferencia principales (Zoom y Teams) que pudieran afectar la integración con el sistema de agendamiento durante el tiempo de desarrollo del proyecto.

4. Se consideró que la demanda de soluciones de videoconferencia seguiría siendo alta, impulsada por la creciente adopción de la educación en línea y la necesidad de soluciones tecnológicas eficientes y seguras.

5. El proyecto se desarrollará bajo la suposición de que cumplirá con todas las normativas legales pertinentes, incluyendo la protección de datos personales y la privacidad, como lo indica la Ley 1581 de 2012 y sus decretos reglamentarios en Colombia.

1.4. Justificación del trabajo de grado

La pandemia de COVID-19 ha catalizado una integración sin precedentes de la educación en línea, resaltando la importancia crítica de plataformas de videoconferencia como Zoom y Teams para mantener la continuidad educativa. Estas herramientas han sido fundamentales para facilitar el aprendizaje remoto en circunstancias desafiantes [1, 2]. Sin embargo, a medida que la educación virtual se consolida como un componente permanente del ámbito educativo, se han evidenciado limitaciones significativas en estas plataformas, particularmente en términos de capacidad y personalización, que pueden comprometer la eficacia de la educación y el aprendizaje en línea [3, 4].

En respuesta a estas limitaciones, esta tesis propone el desarrollo de una arquitectura para un sistema de agendamiento de videoconferencias que integre plataformas existentes y mejore su funcionalidad. El sistema está diseñado para superar las restricciones de capacidad de las plataformas convencionales mediante la automatización de la creación y gestión de múltiples salas de conferencia, permitiendo así acomodar un mayor número de estudiantes sin comprometer la interacción y la calidad del aprendizaje.

Dado que la arquitectura del sistema de agendamiento de videoconferencias requiere una infraestructura que garantice escalabilidad, seguridad y eficiencia, es necesario evaluar qué plataforma en la nube ofrece los servicios más adecuados para su implementación. Una vez que la arquitectura esté definida, se realizará un análisis comparativo entre Amazon AWS y Microsoft Azure, asegurando que la selección de la infraestructura tecnológica esté alineada con los requisitos del sistema diseñado. Optar por una implementación en la nube como Amazon AWS o Microsoft Azure puede reducir significativamente los costos operativos mientras mejora la accesibilidad y la experiencia del usuario. Esta decisión estratégica asegurará que el sistema no solo satisfaga las necesidades actuales sino que también esté preparado para adaptarse a futuros incrementos en la demanda y cambios tecnológicos, contribuyendo a la sostenibilidad y el éxito del sistema en ambientes educativos de gran escala.

Este proyecto no solo busca mejorar la eficiencia y seguridad en el agendamiento de videoconferencias dentro de los entornos educativos, sino que también está alineado con el Objetivo de Desarrollo Sostenible (ODS) número 4, que promueve la Educación de Calidad. A través del desarrollo de una plataforma de agendamiento de videoconferencias que integra sistemas de autenticación y personalización, se pretende garantizar una educación inclusiva y equitativa y promover oportunidades de aprendizaje continuo y accesible para todos.

El sistema propuesto facilitará el acceso a recursos educativos, permitiendo a las instituciones educativas ofrecer una experiencia más adaptable y accesible para estudiantes de diversas procedencias y necesidades. Además, la capacidad del sistema de adaptarse a gran escala contribuye directamente a mejorar la calidad de la educación, permitiendo que educadores y alumnos superen barreras físicas y temporales, y así accedan y participen en procesos de aprendizaje desde cualquier lugar y en cualquier momento.

Capítulo 2

Metodología de la investigación

2.1. Enfoque general y tipo de estudio

La investigación adopta un **diseño mixto secuencial exploratorio–explicativo** [5], idóneo para combinar:

1. la riqueza descriptiva de datos cualitativos y
2. la capacidad de generalización de datos cuantitativos.

Este enfoque se justificó para transitar con rigor desde la problemática detectada en cursos masivos hasta decisiones arquitectónicas fundamentadas en evidencia empírica.

Alcance. El estudio es *aplicado-descriptivo*: persigue caracterizar las limitaciones reales de Zoom/Teams y, a partir de ellas, derivar atributos de calidad (AQ) y lineamientos de diseño sin llegar a construir ni desplegar la solución.

Evaluación arquitectónica sin prototipo. Para mitigar riesgos y obtener evidencia objetiva de que la arquitectura satisfará los atributos de calidad prioritarios, este estudio adopta técnicas de evaluación basadas en escenarios. En particular, el *Architecture Tradeoff Analysis Method* (ATAM) constituye la herramienta principal, mientras que sus predecesores y variantes, SAAM y ARID, respaldan la validez del enfoque [6-8]. Estos métodos permiten analizar decisiones de diseño y documentar riesgos y no-riesgos *antes de escribir una sola línea de código*. Clements *et al.* demuestran que una evaluación produce un conjunto priorizado de escenarios, el mapeo de tácticas a atributos y un registro de riesgos, resultados suficientes para juzgar la idoneidad de la arquitectura sin necesidad de desarrollar un PoC ni un MVP.

2.2. Marco teórico-metodológico y antecedentes

2.2.1. Fundamentos de arquitectura de software

El marco teórico recoge conceptos de *arquitectura de software*, *sistemas distribuidos* e *integración con servicios de terceros*, descritos en el Cap. 3.1. Se enfatizan las directrices de ISO/IEC 25010:2011 para los AQ (*escalabilidad*, *disponibilidad*, *seguridad*) y los métodos Attribute-Driven Design (ADD) y ATAM, utilizados posteriormente como guías de decisión y análisis.

Escenarios como unidad de análisis. Siguiendo a Bass, Clements y Kazman, cada escenario de atributo de calidad se modela con seis partes (fuente, estímulo, ambiente, artefacto, respuesta y métrica de respuesta) [8]. Esta estructura proporciona un lenguaje común entre las distintas comunidades de atributos y cumple para la calidad el mismo rol que los *use cases* cumplen para la funcionalidad, al permitir que las expectativas de los interesados se traduzcan en criterios medibles antes de implementar la solución.

2.2.2. Antecedentes académicos

La revisión sistemática incluyó estudios sobre:

- orquestación de videoconferencias masivas y balanceo de carga en la nube,
- adopción de arquitecturas *serverless* para sistemas educativos,
- comparativas de proveedores IaaS/PaaS respecto de AQ críticos.

Los artículos seleccionados en esta tesis sustentan las decisiones de la Sección 6 y el diseño del Cap. 7.

2.3. Diseño metodológico y fases ejecutadas

La investigación se articuló en cuatro fases secuenciales, cada una asociada a un objetivo específico (véase Figura 2.1):

F1. **Análisis mixto de la encuesta**: se aplicó y trianguló una encuesta Likert con entrevistas semiestructuradas.

F2. **Derivación y priorización de atributos de calidad**: se empleó la matriz MoSCoW sobre los hallazgos de F1.

F3. **Análisis comparativo de nubes**: se compararon AWS y Azure respecto de escalabilidad, disponibilidad, seguridad y coste.

F4. **Diseño arquitectónico**: se utilizó Attribute-Driven Design (ADD) para elaborar la vista C4 y justificar cada decisión.

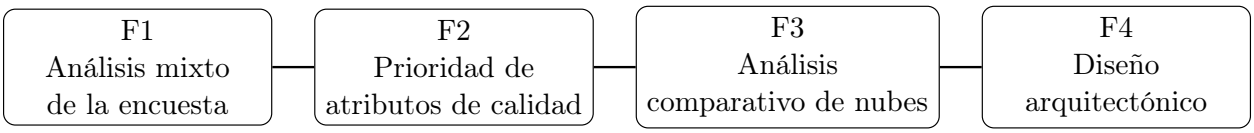


Figura 2.1: Secuencia metodológica alineada a los cuatro objetivos específicos.

2.4. Instrumentos y procedimientos

2.4.1. Instrumento mixto: encuesta y entrevistas

Se empleó un instrumento de naturaleza mixta que combina, en un mismo proceso, la recolección cuantitativa y cualitativa de datos. En primer lugar, se distribuyó una *encuesta Likert* de escala 1–5 cuyos 18 ítems se agrupan en las ocho dimensiones de la norma ISO/IEC 25010; la validez de contenido se garantiza por juicio de expertos y la consistencia interna se verifica mediante el coeficiente α-Cronbach. De forma complementaria, se realizaron *entrevistas semiestructuradas* basadas en un guion de diez preguntas abiertas; las sesiones fueron grabadas, transcritas y codificadas en `Atlas.ti`, lo que permite vincular cada cita a las categorías emergentes y triangular los hallazgos con los resultados numéricos de la encuesta.

2.4.2. Matrices de análisis

- Matriz MoSCoW para priorizar AQ.

- Matriz de comparación de servicios nube (criterios: cobertura funcional, SLA, costos de salida, programas de descuento).

2.4.3. Análisis de trade-offs arquitectónicos (ATAM)

El *análisis ATAM* tomará como insumo directo los artefactos ya documentados en el Cap. 5: el **árbol de utilidad** (Fig. 5.3) y los tres *escenarios de calidad* para Escalabilidad, Disponibilidad y Seguridad (Figs. 5.4, 5.5 y 5.6). Estos escenarios, construidos con la plantilla *Source–Stimulus–Artifact–Environment–Response–Response Measure*, ya incluyen criterios de aceptación cuantificables y sirvieron como hojas del árbol de utilidad, evitando la necesidad de generar nuevos casos durante el taller. Cabe precisar que 'taller' se usa de manera referencial a ATAM: no se efectuaron talleres con stakeholders y los escenarios se establecieron con base en encuestas y análisis propio del autor.

1. **Confirmación de prioridades**. Los participantes revisaron la importancia y la dificultad asignadas a cada hoja del árbol (H/M/L) para validar que siguen reflejando las necesidades actuales de los interesados.

2. **Cobertura y trazabilidad**. Se verifico que cada atributo crítico (escalabilidad, disponibilidad, seguridad) disponga, al menos, de un escenario de uso normal, uno de crecimiento y uno de estrés. Las hojas existentes cumplen ya con esa cobertura básica.

3. **Análisis de tácticas y riesgos**. Para cada escenario se contrastaron las tácticas y patrones propuestos en el Cap. 5, identificando puntos de sensibilidad y trade-offs. El resultado esperado es una lista priorizada de riesgos y no-riesgos que complementará la Sección 5.7.

Siguiendo la guía de Clements *et al.*, los entregables del ATAM (lista de riesgos, puntos de sensibilidad y trade-offs) son suficientes para juzgar la idoneidad de la arquitectura sin necesidad de un PoC ni un MVP.

2.5. Análisis de los datos

2.5.1. Cuantitativo

Medidas descriptivas, prueba t pareada (pre/prioridad–post/prioridad) con $p < 0{,}05$.

2.5.2. Cualitativo

Codificación abierta → axial → selectiva; cálculo de frecuencia y densidad de códigos; ver Sec.4.5.2.

2.5.3. Integración

Triangulación de resultados en tabla de convergencia, estableciendo correspondencia directa entre evidencias y cada AQ.

2.6. Limitaciones del estudio

El análisis se basa en datos autoinformados y en un contexto académico específico; los resultados deben extrapolarse con cautela a entornos corporativos o a otras plataformas de videoconferencia.

2.7. Síntesis

La metodología resume un recorrido coherente desde la revisión teórica hasta el diseño arquitectónico, demostrando cómo los datos empíricos (sin necesidad de implementar la solución) respaldan cada decisión sobre atributos de calidad, patrones arquitectónicos y selección de nube.

Capítulo 3

Marco de referencia

3.1. Marco Teórico

3.1.1. Arquitectura de Software

A lo largo del tiempo se han propuesto múltiples definiciones de *arquitectura de software* por diversos autores e instituciones. En este proyecto adoptamos la definición clásica de Bass *et al.*, quien plantea que la arquitectura de software es "el conjunto de estructuras que se necesitan para razonar sobre un sistema. Estas estructuras comprenden elementos de software, las relaciones entre dichos elementos y las propiedades tanto de los elementos como de las relaciones" [9]. Esta definición enfatiza que la arquitectura de un sistema es una abstracción de alto nivel que expone únicamente aquellos aspectos del software que son significativos para entender su comportamiento y cumplir con sus objetivos, omitiendo detalles irrelevantes [9]. En otras palabras, la arquitectura especifica la organización fundamental de un sistema software, capturando sus componentes principales, cómo interactúan entre sí y bajo qué principios o directrices se diseñó su estructura global [9].

Es importante destacar dos aspectos clave de esta visión: (1) la arquitectura suele involucrar múltiples estructuras o vistas, ya que sistemas complejos pueden requerir diferentes perspectivas (por ejemplo, vista de componentes, vista de despliegue, vista de módulos, etc.) para su completa descripción [10]. (2) La arquitectura constituye las decisiones de diseño más significativas, aquellas que tienen un impacto amplio en la calidad y funcionamiento del sistema. Estas decisiones "arquitectónicamente significativas" incluyen la selección de estilos arquitectónicos, protocolos de comunicación, mecanismos transversales como autenticación, entre otras. Una buena arquitectura sirve como guía para los desarrolladores, facilita la comunicación entre interesados (*stakeholders*) y sienta las bases para lograr los atributos de calidad deseados en el sistema [11]. En síntesis, la arquitectura de software proporciona la estructura organizativa del sistema y actúa como puente entre los requisitos (funcionales y no funcionales) y la implementación concreta del software.

3.1.2. Sistemas Distribuidos

El sistema propuesto para agendamiento de videoconferencias es inherentemente un *sistema distribuido*, dado que involucra múltiples componentes en red (servidores, servicios en la nube, APIs externas) que deben coordinarse para ofrecer la funcionalidad al usuario. En un sistema distribuido, componentes ubicados en nodos separados (generalmente equipos o servicios en la nube distintos)

se comunican y coordinan sus acciones mediante el intercambio de mensajes a través de la red [12]. Esto permite que el procesamiento esté repartido entre varios entornos, mejorando potencialmente la escalabilidad y disponibilidad del sistema, a costa de introducir mayores consideraciones de complejidad en comunicación y consistencia.

En el contexto de esta arquitectura de agendamiento, la distribución implica que diferentes partes del sistema (por ejemplo, el módulo de autenticación, el gestor de calendario, la integración con Zoom, la integración con Teams, la base de datos, etc.) podrían desplegarse en distintos servicios o servidores, comunicándose mediante interfaces bien definidas (como APIs REST). Una arquitectura distribuida bien diseñada puede tolerar fallos parciales (si un componente falla, los demás pueden seguir funcionando), equilibrar la carga de trabajo entre múltiples nodos y acercar ciertos servicios a donde se necesitan (por ejemplo, servicios en la nube geográficamente distribuidos) [13]. No obstante, diseñar sistemas distribuidos requiere atender problemas clásicos como la latencia de red, concurrencia, manejo de transacciones distribuidas, y asegurar la consistencia de datos entre componentes. Es crucial definir protocolos de comunicación robustos y desacoplar componentes para que el fallo de uno no incapacite al resto del sistema. Para lograr esto, suelen emplearse estándares y patrones de integración, tal como se describe en la siguiente sección.

3.1.3. Integración con Servicios de Terceros

Un aspecto central del proyecto es la *integración* con servicios externos de videoconferencia, específicamente Zoom y Microsoft Teams. La integración con terceros consiste en lograr que nuestro sistema se comunique y coordine con plataformas ajenas, típicamente mediante el uso de APIs provistas por esas plataformas. En general, existen patrones de diseño especializados para la integración de sistemas heterogéneos. Un ejemplo ampliamente utilizado es la arquitectura orientada a servicios (*Service-Oriented Architecture*, SOA), donde las funcionalidades se exponen como servicios disponibles en la red mediante protocolos estándar [12]. Bajo SOA (y su evolución moderna a microservicios), es común que los sistemas interactúen usando interfaces web (RESTful APIs o servicios SOAP) independientes del lenguaje o tecnología de implementación. Esto facilita que nuestra aplicación de agendamiento pueda invocar las API de Zoom o Teams para crear reuniones, obtener enlaces, programar eventos, etc., sin necesidad de conocer detalles internos de cada plataforma.

Para lograr una integración efectiva, nuestro diseño adopta principios de la *arquitectura de integración empresarial*. Esto implica usar *conectores* o adaptadores específicos para cada servicio externo y posibles patrones de mensajería. Por ejemplo, se pueden emplear colas de mensajes o mecanismos de *publish-subscribe* para desacoplar la comunicación con terceros, especialmente si algunas interacciones necesitan ser asíncronas [12]. Sin embargo, dado que tanto Zoom como Teams ofrecen APIs REST síncronas (consultas HTTP/JSON típicamente), es viable integrar de forma directa mediante invocaciones HTTP seguras. En todo caso, se recomienda seguir patrones de integración documentados para garantizar confiabilidad y mantenibilidad. Hohpe y Woolf documentan numerosos *Enterprise Integration Patterns* (patrones de integración empresarial) que pueden resultar útiles, tales como *Broker* (intermediario que orquesta llamadas entre productores y consumidores de servicios) o *Facade* (un servicio unificado que encapsula la interacción con múltiples APIs dis-

tintas) [12]. En esta arquitectura, se podría implementar un módulo intermediario que actúe como fachada unificada de "Videoconferencia" y que internamente se comunique con Zoom o Teams según corresponda. Esto encapsula la lógica específica de cada plataforma, reduciendo el acoplamiento del resto del sistema con APIs externas.

Otro aspecto importante de la integración con terceros es manejar las diferencias entre plataformas. Zoom y Teams tienen capacidades y limitaciones particulares (por ejemplo, diferentes parámetros para programar reuniones, distintas formas de gestionar usuarios o autenticación). El sistema debe abstraer estas disparidades presentando a los usuarios una experiencia homogénea. Esto puede implicar transformar y mapear datos entre el formato interno del sistema y el requerido por cada API externa, funcionalidad que típicamente realizan los adaptadores o conectores. Asimismo, es fundamental prever mecanismos de manejo de errores en llamadas remotas, reintentos y registro de eventos, de forma que si un servicio externo está temporalmente no disponible, nuestro sistema pueda reaccionar apropiadamente (por ejemplo, reintentando luego de un intervalo, o notificando al administrador). En resumen, la integración con Zoom y Teams requiere diseñar componentes especializados para comunicación externa, adherir a protocolos estandarizados de la industria y seguir patrones que aseguren un bajo acoplamiento y alta resiliencia en las interacciones entre sistemas.

3.1.4. Computación en la Nube

El uso de *computación en la nube* (*cloud computing*) es un factor clave en la arquitectura propuesta, dado que brinda la base para desplegar un sistema escalable y disponible globalmente. La computación en la nube se define como un modelo de provisión de recursos tecnológicos (como procesamiento, almacenamiento, redes y aplicaciones) bajo demanda, accesibles de forma remota a través de Internet, con un modelo de pago por uso [13]. En lugar de invertir en infraestructura física propia, las organizaciones pueden utilizar servicios en la nube ofrecidos por proveedores como Amazon Web Services, Microsoft Azure o Google Cloud, obteniendo beneficios en elasticidad (es decir, poder escalar recursos según la demanda), flexibilidad y reducción de costos iniciales.

Existen principalmente tres modelos de servicio en la nube ampliamente reconocidos: *Software as a Service* (SaaS), *Platform as a Service* (PaaS) e *Infrastructure as a Service* (IaaS) [13]. En el contexto de nuestro sistema, SaaS se refiere a utilizar aplicaciones completas ofrecidas por terceros (por ejemplo, Zoom y Teams mismas son servicios SaaS de videoconferencia en la nube). PaaS ofrece plataformas administradas (entornos de ejecución, bases de datos, etc.) donde podemos desplegar nuestra aplicación de agendamiento sin preocuparnos por la gestión del sistema operativo o hardware subyacente. IaaS proporciona infraestructura virtual (máquinas virtuales, almacenamiento, redes) sobre la cual uno mismo configura y despliega los componentes necesarios. Para nuestro diseño integrado, podríamos aprovechar una combinación de estos modelos: por un lado consumimos los SaaS de videoconferencia (Zoom/Teams) a través de sus APIs, y por otro lado podríamos desplegar la lógica propia de la aplicación de agendamiento en una plataforma PaaS (por ejemplo, Azure App Service, AWS Elastic Beanstalk, etc.) o incluso como microservicios en contenedores orquestados (usando Kubernetes en la nube, por ejemplo).

El aprovechamiento de la nube nos aporta varios atributos de calidad importantes. Uno de ellos

es la **disponibilidad**: los principales proveedores de nube permiten replicar servicios en múltiples zonas geográficas y ofrecen balanceo de carga y redundancia, lo que significa que nuestra aplicación puede mantenerse disponible incluso si ocurren fallos en un centro de datos [13]. Otro atributo es la **escalabilidad**: la nube facilita escalar verticalmente (más recursos en un servidor) u horizontalmente (más instancias de servidores) de forma automática según la carga del sistema [13]. Esto es esencial para un sistema multi-sala que podría experimentar picos de uso (por ejemplo, al inicio de horas de clase en un campus, cuando muchas videoconferencias se inician simultáneamente). Adicionalmente, la nube ofrece **interoperabilidad** y servicios gestionados: por ejemplo, podríamos utilizar servicios de base de datos, colas de mensajería, funciones serverless, que se integran fácilmente con nuestras aplicaciones y permiten construir una arquitectura distribuida más robusta con menor esfuerzo de administración.

Cabe mencionar que el despliegue en la nube también conlleva consideraciones de diseño: se debe planificar la configuración de *autoscaling*, la tolerancia a fallos, la distribución geográfica adecuada según los usuarios (para minimizar latencia), y la seguridad en la nube (configuración correcta de redes virtuales, cortafuegos, credenciales para acceder a APIs, etc.). En síntesis, la computación en la nube proporciona el cimiento tecnológico para implementar nuestra arquitectura de manera eficiente, escalable y altamente disponible, alineada con las necesidades de un sistema integrado de agendamiento de videoconferencias.

3.1.5. Atributos de Calidad

Los *atributos de calidad* (también conocidos como requisitos no funcionales) son propiedades medibles o comprobables de un sistema software que describen *cómo* se comporta el sistema o cuán bien realiza ciertas funciones, en contraste con los requisitos funcionales que describen *qué* hace el sistema. Ejemplos típicos de atributos de calidad son la seguridad, el rendimiento, la disponibilidad, la escalabilidad, la interoperabilidad, la usabilidad, entre otros [14]. La relevancia de los atributos de calidad en el diseño de la arquitectura es fundamental: una buena arquitectura se orienta no solo a satisfacer las funcionalidades requeridas, sino a garantizar que el sistema cumpla ciertos niveles de desempeño, seguridad, flexibilidad y demás aspectos de calidad que son críticos para los interesados (*stakeholders*) [11].

La norma internacional ISO/IEC 25010:2011 establece un modelo de calidad del producto software que agrupa los atributos de calidad en ocho grandes categorías: adecuación funcional, eficiencia de desempeño, compatibilidad, usabilidad, fiabilidad, seguridad, mantenibilidad y portabilidad [14]. Cada categoría se descompone a su vez en subatributos específicos. Por ejemplo, la categoría de **fiabilidad** incluye subatributos como disponibilidad, tolerancia a fallos y capacidad de recuperación; la **compatibilidad** abarca la interoperabilidad; la **eficiencia de desempeño** cubre tiempos de respuesta, uso de recursos y capacidad, etc. [14]. En el contexto de nuestro sistema multi-sala de videoconferencias, varios atributos de calidad son especialmente relevantes: • Disponibilidad: dado que el sistema gestionará reuniones en vivo para usos académicos, debe estar disponible de forma casi ininterrumpida, especialmente en horarios críticos. Cualquier indisponibilidad impediría iniciar o programar reuniones, afectando las actividades universitarias. Por tanto, la arquitectura debe mi-

nimizar puntos únicos de fallo y soportar conmutación o recuperación rápida [13]. • Escalabilidad: el sistema debe soportar potencialmente decenas o cientos de salas virtuales simultáneas. La capacidad de escalar implica que el rendimiento no se degrade significativamente a medida que aumenta la carga (número de solicitudes, usuarios concurrentes, etc.) [13]. Este atributo se logra mediante diseño eficiente (p.,ej., uso de balanceo de carga, instancias adicionales en la nube bajo demanda, y una correcta gestión de recursos). • Interoperabilidad: es la habilidad del sistema de funcionar con otros sistemas. En nuestro caso, se refiere a la capacidad de integrarse con plataformas de videoconferencia heterogéneas (Zoom, Teams) y posiblemente con sistemas internos de la universidad (calendarios institucionales, sistemas de gestión académica, etc.). Una alta interoperabilidad significa que el sistema utiliza protocolos y formatos estándar para comunicarse (como REST/JSON, OAuth 2.0 para autenticación, etc.), facilitando la comunicación fluida con los servicios externos [14]. • Seguridad: abarca varios aspectos, desde la confidencialidad (protección de datos sensibles como credenciales de cuentas), integridad (evitar modificaciones indebidas en la información de reuniones) hasta autenticación y autorización apropiadas (asegurarse de que solo usuarios legítimos puedan programar/ingresar a las videoconferencias). Dado que el sistema actuará como intermediario entre usuarios y plataformas externas, debe manejar token de acceso, contraseñas de reuniones y otra información sensible de manera segura (cifrado en tránsito y en reposo, políticas de expiración de sesiones, etc.) [15]. • Usabilidad: aunque es un atributo orientado al usuario final, merece atención para asegurar que la herramienta de agendamiento sea fácil de utilizar por parte de docentes y administrativos. Esto implica una interfaz clara para programar sesiones en distintas plataformas sin requerir conocimientos técnicos avanzados. Una buena usabilidad reduce errores de uso y mejora la adopción del sistema.

En la metodología de diseño adoptada (ADD, ver sección siguiente), los atributos de calidad se tratan como *Drivers de Arquitectura* [11]. Es decir, desde el inicio del proceso de diseño se identifican cuáles atributos de calidad son críticos para el éxito del sistema y se elaboran escenarios concretos para esos atributos. Un *escenario de calidad* tipifica una situación específica (estímulo, entorno, respuesta esperada) que el sistema debe manejar; por ejemplo: "*ante 100 solicitudes concurrentes de programación de reuniones (estímulo, carga máxima), el sistema debe confirmarlas en menos de 2 segundos (respuesta de desempeño) mientras mantiene la integridad de datos y sin rechazar peticiones (medida de resiliencia)*". Estos escenarios permiten evaluar si las decisiones arquitectónicas propuestas satisfarán las expectativas de calidad. Por lo anterior, los atributos de calidad son un pilar fundamental del marco teórico del diseño arquitectónico, ya que guían las decisiones estructurales y la selección de patrones o tácticas apropiadas para cumplir con las exigencias no funcionales del sistema [11].

3.1.6. Diseño Orientado por Atributos (ADD)

Para abordar sistemáticamente el diseño arquitectónico enfocándose en los atributos de calidad, se emplea la metodología *Attribute-Driven Design* (ADD). ADD es un enfoque iterativo desarrollado por el Software Engineering Institute (SEI) de Carnegie Mellon, que propone diseñar la arquitectura basándose primordialmente en los requisitos de calidad del software [11]. En lugar de partir

únicamente de la funcionalidad, ADD comienza identificando los atributos de calidad más importantes (según las necesidades del negocio y los usuarios) y luego guía la definición de la estructura arquitectónica para asegurar que dichos atributos sean satisfechos.

El proceso de ADD se puede resumir en una secuencia cíclica de "planear, realizar y verificar" [11]. En la fase de **planificación**, se seleccionan y priorizan los escenarios de calidad y se establecen restricciones de diseño iniciales. Esta priorización (p. ej., determinar qué atributos de calidad son "alto", "medio" o "bajo" prioridad) orienta qué concerns deberán ser abordados primero en la arquitectura propuesta. A continuación, en la etapa de **diseño** (realización), se eligen tácticas y patrones arquitectónicos apropiados para cumplir con los escenarios de calidad priorizados. Una *táctica* arquitectónica es una técnica de diseño específica para influir en un atributo de calidad particular [11]. Por ejemplo, si la disponibilidad es crítica, se puede aplicar la táctica de redundancia (componentes duplicados en espera de fallo); si el rendimiento es crucial, se pueden emplear tácticas de caché o reducción de comunicaciones remotas, etc. Estas tácticas se combinan en una estructura (conjunto de componentes, conectores y sus interacciones) que constituye un primer boceto de la arquitectura.

Posteriormente, en la fase de **verificación**, se analiza y evalúa el diseño resultante contra los escenarios de calidad definidos, para comprobar si las decisiones tomadas efectivamente permiten alcanzar los niveles esperados de calidad [9]. Si se encuentran brechas o posibles mejoras, el proceso itera nuevamente: se refinan los escenarios (o se abordan los siguientes en prioridad), se hacen ajustes al diseño incorporando nuevas tácticas o patrones, y se vuelve a verificar. Este proceso iterativo continúa hasta que la arquitectura diseñada satisface adecuadamente los requerimientos tanto funcionales como de calidad [9].

Una fortaleza de ADD es que integra desde el inicio la consideración explícita de las **restricciones de diseño** impuestas (por ejemplo, "el sistema debe implementarse con ciertos componentes ya existentes" o "debe desplegarse en una nube específica"). Estas restricciones, junto con los requisitos funcionales y los atributos de calidad, constituyen los *drivers* de diseño que en cada iteración de ADD definen el problema de diseño a resolver [11]. Por tanto, ADD ayuda a descomponer recursivamente el sistema: en la primera iteración se define la estructura de alto nivel (división en subsistemas o módulos principales) con base en las preocupaciones principales de calidad; en iteraciones sucesivas se refina el diseño de cada subsistema, aplicando nuevamente tácticas para atributos de calidad a nivel más detallado.

En este proyecto, ADD sirve como la metodología guía para diseñar la arquitectura integrada de nuestro sistema multi-sala. En particular, dado que atributos como disponibilidad, interoperabilidad y seguridad son críticos, ADD orienta la selección de patrones (por ejemplo, usar microservicios para aislar funciones y facilitar escalabilidad, emplear un servicio de autenticación centralizado, incluir un balanceador de carga para alta disponibilidad, etc.) en coherencia con esos atributos. El resultado del proceso ADD es una especificación arquitectónica donde cada decisión importante puede trazarse a un requerimiento de calidad o restricción que la motivó, asegurando así que la arquitectura final esté alineada con las necesidades del proyecto [9]. En síntesis, el diseño orientado por atributos proporciona un marco riguroso para construir arquitecturas robustas frente a las demandas de calidad, minimizando la posibilidad de que aspectos críticos (como el rendimiento o la seguridad) sean descubiertos tardíamente o resueltos ad-hoc sin un enfoque sistemático.

3.1.7. Patrones Arquitectónicos

Los *patrones arquitectónicos* son soluciones probadas y generalizadas para problemas recurrentes en el diseño de software a gran escala [9]. Actúan como plantillas de alto nivel que indican cómo organizar los componentes y las interacciones en un sistema. Al iniciar el diseño de una arquitectura, es habitual seleccionar uno o varios patrones que servirán de base estructural, según la naturaleza del sistema y los requisitos (funcionales y de calidad) que deban satisfacerse [9].

Existen numerosos patrones arquitectónicos documentados en la literatura. A modo de ejemplo, Bass *et al.* [9] describen una familia de patrones clásicos, entre los cuales destacan:

- **Patrón en Capas (Layered):** Organiza el sistema en capas jerárquicas, donde cada capa ofrece servicios a la capa superior y consume los servicios de la capa inferior. Es útil para lograr separación de preocupaciones y facilitar la mantenibilidad. En nuestro sistema, podríamos pensar en capas como: interfaz de usuario, lógica de negocio (agenda, reglas de disponibilidad), y capa de integración con servicios externos.

- **Patrón Cliente-Servidor:** Distingue entre clientes (que realizan peticiones de servicio) y servidores (que responden a esas peticiones). La web y muchas arquitecturas distribuidas se basan en este modelo. Nuestra aplicación de agendamiento actuará en parte como cliente (cuando invoca a Zoom/Teams) y en parte como servidor (cuando brinda funcionalidades a los usuarios finales).

- **Patrón Broker (Intermediario):** Introduce un componente mediador que coordina la comunicación entre múltiples clientes y servidores distribuidos. Un *broker* registra y localiza servicios, y enruta las solicitudes desde clientes a las instancias apropiadas de servidor [9]. Esto es relevante si imaginamos que podríamos tener múltiples instancias de servicios (por ejemplo, varios módulos que manejan videoconferencias en Zoom o Teams) y queremos una capa que direccione las peticiones al módulo correcto.

- **Patrón Pipe-and-Filter:** Establece una cadena de componentes (*filtros*) donde la salida de uno es la entrada del siguiente, conectados por canales (*pipes*). Se usa en procesamiento secuencial de datos. Quizás menos aplicable directamente a nuestro caso, aunque podríamos ver el flujo de creación de una reunión como pasando por etapas (validación, reserva de sala, programación en plataforma) que podrían modelarse con este patrón.

- **Patrón Publica-Suscribe:** Los componentes productores *publican* eventos o mensajes en un canal común, y los consumidores *se suscriben* a los tipos de eventos que les interesan. Este patrón es útil para notificaciones y coordinación asíncrona [12]. Podría utilizarse, por ejemplo, si deseamos que cuando se programe una reunión en nuestro sistema, otros subsistemas (como un calendario general de la universidad, o un sistema de notificación por correo) reciban el aviso automáticamente.

- **Patrón Orientado a Servicios (SOA):** Como ya se mencionó, en SOA la computación se organiza en torno a servicios autónomos que interactúan a través de una red, típicamente des-

cribiendo flujos de trabajo de más alto nivel. La arquitectura de microservicios es una especialización moderna de SOA donde cada servicio es muy granular y autónomo. En nuestro caso, podemos concebir una arquitectura de microservicios donde módulos separados gestionen, por ejemplo, la orquestación de Zoom, la orquestación de Teams, la autenticación de usuarios, la gestión de reservas de salas físicas, etc., comunicándose entre sí mediante APIs internas [9]. Los microservicios permiten escalar y desplegar cada módulo de forma independiente, mejorando la disponibilidad y tolerancia a fallos del conjunto [16]. No obstante, requieren mecanismos de orquestación (por ejemplo, un *gateway* API, descubrimiento de servicios, monitoreo distribuido) que añaden complejidad.

La selección de patrones arquitectónicos se realiza en función de los *drivers* del diseño (requisitos y atributos de calidad). Por ejemplo, si la **modificabilidad** y **mantenibilidad** fueran primordiales, un patrón en capas bien definido ayudaría a aislar cambios en una capa sin impactar las otras [9]. Si la prioridad es la **escala masiva** y **despliegue flexible**, una arquitectura de microservicios en contenedores podría ser la elección, combinada con patrones de infraestructura como *autoescalado* y *balanceo de carga* [13]. En nuestro caso particular, donde la **interoperabilidad** con servicios externos y la **disponibilidad** son cruciales, patrones como SOA/microservicios y broker resultan muy adecuados: SOA/microservicios nos da modularidad y posibilidad de escalamiento independiente por componente, mientras que un broker (o gateway) de integración centralizado puede simplificar la comunicación con múltiples plataformas de videoconferencia. Adicionalmente, para reforzar disponibilidad, es común aplicar el patrón de *redundancia* (múltiples instancias de servicios críticos detrás de un balanceador, de modo que si una instancia falla otra tome su lugar) [13]. Aunque la redundancia en sí no es un patrón de arquitectura de software, es una táctica de calidad que suele implementarse apoyándose en la infraestructura de nube (p. ej., usando múltiples zonas de disponibilidad).

En resumen, los patrones arquitectónicos proveen esquemas reconocidos para estructurar nuestro sistema. Al apoyarnos en ellos, nos beneficiamos de la experiencia previa en sistemas similares, reduciendo riesgos de diseño. No obstante, es frecuente que un sistema real combine varios patrones a diferentes niveles; por ejemplo, podríamos tener un patrón global SOA con microservicios, y dentro de cada microservicio aplicar MVC u otro patrón para su diseño interno. Lo importante es que la arquitectura resultante satisfaga las necesidades del sistema y cumpla con los atributos de calidad esperados, y los patrones son herramientas conceptuales que nos ayudan a lograr ese objetivo de manera organizada y comprensible [9].

3.1.8. Autenticación y Seguridad en la Integración

La *autenticación* es un componente transversal vital en un sistema que integra servicios de terceros, especialmente cuando estos servicios (Zoom, Teams) requieren acceso a recursos protegidos (por ejemplo, programar reuniones en la cuenta de un usuario, obtener listas de participantes, etc.). Para interactuar con las APIs de Zoom y Microsoft Teams de forma segura, se debe seguir un protocolo de autenticación estándar. Actualmente, el mecanismo más utilizado es **OAuth 2.0**, un marco de autorización que permite a nuestra aplicación obtener permisos limitados para actuar en

nombre del usuario sin necesidad de conocer sus credenciales (contraseña) [15].

En el flujo típico de OAuth 2.0, cuando un usuario de nuestro sistema quiere vincular su cuenta de Zoom o Teams para agendar reuniones, se le redirige al proveedor (Zoom/Teams) para que ingrese sus credenciales y otorgue consentimiento. Una vez autorizado, el proveedor devuelve a nuestra aplicación un *token de acceso* (y posiblemente un *token de refresco*) que nuestra aplicación usará en adelante para realizar peticiones autenticadas a la API del servicio externo [15]. Este token es esencialmente una llave temporal que el sistema debe almacenar de forma segura (idealmente cifrada) y usarla al invocar, por ejemplo, "crear reunión en Zoom" o "programar evento en Teams" enviándola en la cabecera de autorización. Los tokens de acceso suelen tener expiración corta (por ejemplo, una hora), tras lo cual es necesario usar el token de refresco para obtener uno nuevo sin molestar nuevamente al usuario [15].

Adicionalmente, nuestra arquitectura debe implementar correctamente el esquema de *roles y autorizaciones* interno. Es decir, no solo autenticamos contra servicios externos, sino que también autenticamos a los usuarios contra nuestro propio sistema. Probablemente integremos el inicio de sesión institucional de la universidad o un directorio de usuarios existente. Una vez que sepamos quién es el usuario dentro de nuestro sistema, debemos controlar qué operaciones puede hacer (por ejemplo, un profesor podrá programar reuniones asociadas a sus clases, un administrador podría ver métricas globales, etc.). Esta gestión de roles y permisos es parte del diseño de seguridad interno.

Otro aspecto a considerar es la **seguridad en la comunicación**. Todas las interacciones con APIs de terceros deben realizarse sobre canales cifrados (HTTPS/TLS) y verificando la identidad de los servidores mediante certificados válidos, algo que por defecto ya se consigue si usamos las bibliotecas web estándar. Asimismo, al almacenar cualquier dato sensible (tokens OAuth, identificadores de reuniones, etc.), debemos protegerlos en nuestra base de datos. Por ejemplo, no conviene almacenar *plain tokens* indefinidamente; una buena práctica es encriptarlos con una clave mantenida segura, o aprovechar almacenes de secretos de la nube (p. ej., AWS Secrets Manager, Azure Key Vault) para guardar credenciales de integración [13].

La integración con plataformas como Microsoft Teams puede involucrar protocolos adicionales. En entornos empresariales, a veces se usa *Single Sign-On* (SSO) con SAML o OpenID Connect para autenticar usuarios de forma federada. En nuestro caso, podríamos permitir que el usuario ingrese a nuestro sistema con sus credenciales de Office 365 (usando OAuth/OpenID Connect de Microsoft), lo que simultáneamente otorgaría permisos para Teams. Son posibilidades de diseño que mejorarían la experiencia de usuario (menos contraseñas separadas) pero requieren coordinar con la infraestructura de identidad de la universidad.

En resumen, la arquitectura debe garantizar que **sólo usuarios autorizados** puedan acceder a las funciones de agendamiento y que **sólo peticiones legítimas** se envíen a las APIs externas con los permisos adecuados. OAuth 2.0 nos proporciona un marco robusto para lo segundo, delegando en Zoom y Teams la autenticación del usuario y recibiendo tokens de acceso limitados [15]. Internamente, adoptaremos el principio de mínimo privilegio: cada componente o servicio de nuestro sistema (p. ej., un microservicio de integración con Zoom) solo manejará los tokens y datos necesarios para su función, y las credenciales maestras (como secretos de API) se guardarán de forma segura. Adicionalmente, es recomendable registrar todas las operaciones relevantes en un *log*

de auditoría (por ejemplo, quién programó qué reunión y cuándo, si hubo intentos fallidos de acceso, etc.) para poder monitorear y detectar usos indebidos o anómalos.

En términos de atributos de calidad, todo lo anterior se relaciona con la **seguridad** y también con la **fiabilidad** del sistema. Un sistema más seguro es a la vez más confiable para los usuarios y las organizaciones que confían en él. Implementar correctamente la autenticación y la autorización asegura la *integridad* y *confidencialidad* de los datos manejados, lo cual en un entorno académico protege información sensible (p. ej., enlaces privados de reuniones, grabaciones, etc.). Por tanto, la seguridad no es un componente aislado, sino que permea todo el diseño arquitectónico: desde la comunicación entre módulos, el acceso a terceros, hasta el almacenamiento y presentación de la información.

3.1.9. Escalabilidad

La *escalabilidad* es la capacidad del sistema para manejar crecientes volúmenes de trabajo (usuarios, operaciones, datos) aumentando su rendimiento de forma proporcional, usualmente mediante la adición de recursos [13]. En una solución de agendamiento multi-sala, la escalabilidad es un atributo crítico, ya que en un campus universitario grande podría haber picos con decenas de reuniones iniciando simultáneamente a ciertas horas, o cientos de usuarios consultando horarios. El sistema debe poder escalar para atender esas cargas pico sin degradación significativa en la experiencia de usuario.

Existen dos enfoques principales de escalabilidad: **escalamiento vertical** (dotar de más capacidad a un solo servidor, por ejemplo, más CPU/RAM) y **escalamiento horizontal** (añadir más instancias de servidor que trabajen en paralelo). Las arquitecturas modernas en la nube privilegian el escalamiento horizontal, ya que permite crecer prácticamente sin límite simplemente aprovisionando instancias adicionales, y además mejora la tolerancia a fallos (si una instancia falla, otras siguen sirviendo) [13]. En nuestro diseño, gracias a la adopción de una arquitectura modular (p. ej., microservicios o al menos separación de componentes), podemos escalar independientemente las partes del sistema que presenten mayor demanda. Por ejemplo, si el módulo de integración con Zoom soporta más carga que el de Teams, podríamos aumentar solo las instancias del primero.

La escalabilidad también se ve beneficiada por patrones específicos. Uno muy común es el uso de **balanceadores de carga** (*load balancers*), que distribuyen automáticamente las solicitudes entrantes entre varias instancias de servicio equivalentes [13]. Colocar un balanceador de carga al frente de nuestro servicio web de agendamiento permite repartir las peticiones de los usuarios entre múltiples servidores de aplicación. A nivel de integración con terceros, si tenemos múltiples instancias manejando peticiones a Zoom/Teams, también deben coordinarse sin que ninguna instancia individual se sobrecargue.

Otro aspecto de la escalabilidad es el manejo eficiente de los recursos: por ejemplo, mantener sesiones livianas, usar caches de resultados para evitar llamadas repetitivas costosas, liberar correctamente conexiones a APIs externas, etc. Una práctica recomendada es implementar *limitación de carga* (*throttling*) y *pools de conexiones* para no exceder los límites de las APIs de terceros (que suelen imponer cuotas de llamadas por minuto/hora). Así, nuestro sistema debe escalar no solo

internamente, sino hacerlo de forma consciente de que depende de servicios externos con limitaciones. Esto podría implicar, por ejemplo, encolar internamente ciertas solicitudes de agendamiento si superan momentáneamente la capacidad de invocar a Zoom/Teams, procesándolas con un leve retardo pero asegurando que eventualmente se realicen sin fallar.

En entornos cloud, lograr escalabilidad a menudo se automatiza con servicios de *auto-escalado*: configuraciones que monitorean métricas (CPU, número de peticiones, tiempos de respuesta) y automáticamente lanzan instancias adicionales cuando se superan umbrales, o las apagan cuando la demanda baja [13]. Podemos configurar reglas de auto-escalado para nuestro backend de agendamiento de manera que, por ejemplo, de 8am a 5pm (horario de clases) haya un número mínimo de instancias preparadas, incrementándose si el CPU excede cierto porcentaje con picos de uso.

La **arquitectura orientada a microservicios** mencionada previamente es especialmente favorable para la escalabilidad, porque cada microservicio puede escalarse individualmente. Por ejemplo, si el componente de notificaciones en tiempo real (avisos de que una reunión va a empezar) se vuelve un cuello de botella, podemos replicarlo varias veces, sin necesidad de escalar también los otros componentes que quizá no lo necesiten [16]. Esto evita consumir recursos de más en partes poco usadas, optimizando costos en la nube.

Finalmente, cabe relacionar la escalabilidad con la **disponibilidad**: un sistema bien escalado evita sobrecargas que pueden tumbar servicios, por tanto mejora su uptime. La arquitectura deberá también considerar escenarios de *scaling out/in* rápidos para atender subidas bruscas de carga (por ejemplo, al comienzo de un semestre podría haber muchas configuraciones nuevas). Un diseño sin estado (*stateless*) en los componentes de negocio facilitará esto, ya que las instancias pueden añadirse o removerse sin necesidad de migrar información de sesión. En caso de que haya estado (p. ej., sesiones de usuario), se suelen externalizar en almacenes compartidos (como Redis para sesiones), para que cualquier instancia pueda atender cualquier usuario.

En conclusión, la escalabilidad se asegura combinando: elección adecuada de patrones (microservicios, SOA, etc.), infraestructura cloud con auto-escalado y balanceo de carga, tácticas de optimización de desempeño (cachés, procesamiento asíncrono, etc.), y pruebas de carga rigurosas para validar que el sistema se comporta correctamente al incrementarse la demanda. Este conjunto de consideraciones permite que el Sistema de Agendamiento Multi-Sala pueda crecer en alcance (más usuarios, más reuniones, más sedes o campus) sin necesidad de rediseños fundamentales ni caídas en su nivel de servicio, cumpliendo así con uno de los objetivos primordiales de su arquitectura.

3.2. Antecedentes

3.2.1. Antecedentes Académicos

En el ámbito académico y de la investigación en ingeniería de software, se han desarrollado numerosos trabajos relacionados con la construcción de arquitecturas integradas, el uso de plataformas de videoconferencia y la gestión de sistemas distribuidos similares al que aquí se propone. A continuación, se resumen algunos antecedentes relevantes que proporcionan contexto y soporte conceptual a este proyecto.

Con la expansión de la educación virtual y las necesidades surgidas durante la pandemia de COVID-19, varias investigaciones han comparado y evaluado las principales plataformas de videoconferencia disponibles. Por ejemplo, **Chawla et al. (2021)** presentan "A Tale of Three Videoconferencing Applications: Zoom, Webex and Meet", un análisis comparativo de estas plataformas donde se discuten características, desempeño y usabilidad [17]. Sus hallazgos resaltan las fortalezas de Zoom en estabilidad de conexión y facilidad de uso, mientras que destacan la integración nativa de Google Meet con otras herramientas de Google Workspace, y las capacidades de seguridad y control administrativo en Webex. Aunque Microsoft Teams no fue parte de esa comparación específica, otros estudios han enfocado Teams frente a Zoom.

En particular, se han evaluado estas plataformas en contextos educativos. **Moorhouse & Kohnke (2021)** realizaron un estudio sobre las funcionalidades de Zoom y cómo influyen en la dinámica de participación en clases en línea [18], señalando que ciertas características (p.ej., compartir pantalla, salas de descanso) bien utilizadas pueden mejorar la interacción estudiante-docente, pero también identificando retos como la *fatiga de Zoom* en sesiones prolongadas. Por su parte, **Naveh et al. (2021)** llevaron a cabo una comparación de plataformas de videoconferencia (incluyendo Zoom, Teams, Google Meet) específicamente desde la experiencia de usuarios con discapacidades, evaluando la accesibilidad y las facilidades que cada plataforma ofrece [19]. Este tipo de investigaciones nos indican que ambas plataformas de nuestro interés (Zoom y Teams) son ampliamente aceptadas pero tienen diferencias en experiencia de usuario y en su eco-sistema de integración.

Otro cuerpo de trabajo relevante es aquel relacionado con la **usabilidad y adopción** de herramientas de videoconferencia en entornos universitarios. **Gómez *et al.* (2022)** presentaron un estudio de caso sobre el impacto del uso de Microsoft Teams en la enseñanza y el aprendizaje en una universidad (WSU Butterworth Campus) [20]. Los resultados muestran que, tras superar la curva de aprendizaje inicial, Teams ofreció mejoras en colaboración y comunicación institucional, aunque la saturación de notificaciones y la organización de equipos supuso desafíos menores para algunos usuarios. Estos hallazgos justifican la importancia de contar con una herramienta integradora como la que proponemos: dado que Zoom y Teams ofrecen diferentes ventajas, muchas instituciones optan por utilizarlas de forma complementaria. Un sistema que unifique la programación de reuniones en ambas podría potenciar ese beneficio dual, aliviando las dificultades de gestión al tener múltiples plataformas.

En cuanto a arquitectura de software, existen antecedentes de proyectos que aplican metodologías similares a ADD y que se orientan a resolver problemas de integración o escalabilidad. Por ejemplo, el trabajo de **Palacios (2024)** diseñó una arquitectura de software en la nube de AWS para la

empresa Italcol, utilizando precisamente la metodología de Diseño Orientado por Atributos para priorizar patrones de rendimiento y seguridad [21]. Aunque el dominio de aplicación es distinto (una empresa de manufactura), la aproximación sirve de referencia metodológica: establecieron escenarios de calidad (como alta disponibilidad, interoperabilidad entre sistemas de la empresa y de socios) y eligieron soluciones arquitectónicas alineadas (microservicios desplegados en AWS, colas de mensajería para integrar sistemas internos, autenticación centralizada, etc.). Este enfoque es análogo al que empleamos en nuestro proyecto, donde también la interoperabilidad (entre nuestra aplicación y servicios de videoconferencia) y la disponibilidad son ejes centrales.

En el entorno universitario local, **Cruz (2023)** implementó una herramienta de línea de producción de software (*pipeline* CI/CD) que integraba múltiples servicios en la nube (Azure DevOps, GitHub, Jenkins, etc.), con el fin de automatizar despliegues [22]. Si bien se trata de un dominio distinto (DevOps), conceptualmente es un antecedente de integración multi-plataforma: su solución debía orquestar diferentes servicios terceros de forma unificada. La tesis de Cruz destaca la importancia de una capa de abstracción común y de adaptadores para cada plataforma, algo que en nuestro proyecto es directamente aplicable (una capa de orquestación general de videoconferencias, con adaptadores específicos para Zoom y Teams). Además, en su evaluación, los desarrolladores valoraron positivamente tener una sola herramienta que gestionara diferentes servicios, en lugar de interactuar manualmente con cada uno. Por analogía, esperamos que nuestra herramienta de agendamiento centralizado aporte simplicidad y eficiencia a los usuarios al evitarles alternar entre distintas aplicaciones según la plataforma de reunión.

3.2.2. Proyectos Similares y Soluciones Existentes

En cuanto a soluciones ya existentes en el mercado o en entornos universitarios para el problema de agendar videoconferencias en múltiples plataformas, cabe señalar que actualmente muchas instituciones han adoptado integraciones parciales mediante complementos o extensiones. Por ejemplo, existen *plug-ins* para sistemas de gestión de aprendizaje (LMS) como Moodle o Canvas que permiten programar reuniones Zoom desde el calendario del LMS. Igualmente, Microsoft Teams se ha integrado con algunas plataformas educativas, ofreciendo la creación de reuniones desde las propias aplicaciones institucionales. Sin embargo, una solución genérica que combine ambas plataformas en un solo scheduler no es común en forma de producto comercial unificado.

No obstante, sí encontramos herramientas que abordan partes del problema. Zoom ofrece un **Marketplace de Apps** con aplicaciones de terceros: por ejemplo, hay aplicaciones que sincronizan eventos de Zoom con Google Calendar u Outlook. Microsoft Teams, a través de Microsoft Graph API, también permite que aplicaciones externas creen eventos de calendario que tienen enlaces Teams. Estos mecanismos indican que, técnicamente, es viable integrar ambas plataformas con un sistema central, reforzando la factibilidad de nuestro enfoque. Un reto habitual mencionado en foros técnicos es mantener la **consistencia** entre plataformas: si un usuario cambia la hora de una reunión en Zoom, ¿cómo reflejar ese cambio en la entrada correspondiente de Teams (si es que existiera)? Nuestra arquitectura deberá considerar cómo manejar eventos de actualización o cancelación de reuniones de forma coherente, quizás definiendo que el sistema central sea la “fuente de verdad” y

sincronice cambios a las plataformas.

En términos de iniciativas académicas similares, algunos campus han desarrollado scripts o pequeñas aplicaciones para facilitar a profesores la creación de sus enlaces de clases virtuales. Sin embargo, suelen estar limitadas a una sola plataforma (por ejemplo, toda la institución usa Zoom, o usa Teams exclusivamente). El escenario multi-plataforma añade complejidad pero también flexibilidad. Dado que tanto Zoom como Teams tienen APIs robustas y bien documentadas, este proyecto se beneficia de los **SDKs y servicios** ya disponibles: por ejemplo, Zoom API permite no solo crear y eliminar reuniones, sino también obtener reportes de asistencia, grabaciones, etc., lo cual podría potencialmente extender las capacidades de nuestro sistema a futuro (integrar informes, etc.). Microsoft Graph API para Teams permite crear reuniones en el calendario de un usuario y obtener enlaces de *Meetings* con opciones configurables (p. ej., si la sala de espera está habilitada, si se graba automáticamente, etc.).

Por último, es pertinente mencionar tendencias actuales que si bien no abordan exactamente el mismo problema, sí lo tangencian: las llamadas **plataformas de colaboración unificada**. Empresas tecnológicas ofrecen suites que engloban chat, videoconferencia, calendario y gestión de proyectos en un solo ecosistema (por ejemplo, Microsoft Teams intenta ser eso dentro del universo Microsoft). La visión de nuestro proyecto es distinta en cuanto busca una integración "horizontal" entre dos ecosistemas líderes, pero es compatible con la noción de simplificar la experiencia del usuario final. En un entorno donde conviven Zoom y Teams, nuestra solución aspira a ser esa capa unificadora que dé una apariencia de plataforma unificada al usuario, aunque por detrás esté coordinando dos infraestructuras distintas.

En síntesis, los antecedentes muestran que:

- Zoom y Teams han sido ampliamente estudiados y utilizados en educación; cada uno con ventajas, lo que respalda la utilidad de soportar ambos.

- Existen metodologías de diseño (ADD) y patrones probados que proyectos previos han aplicado con éxito para arquitecturas integradas y en la nube.

- No se identifica una herramienta previa exactamente igual a la propuesta, lo que confirma la originalidad y necesidad del trabajo, pero sí existen componentes y APIs disponibles que facilitan su realización.

Estos antecedentes sirven como guía y validación tanto de los requerimientos de nuestro sistema como de las decisiones de diseño arquitectónico tomadas, asegurando que la propuesta se alinea con prácticas recomendadas y cubre un vacío real en el entorno de las herramientas de colaboración académica.

Capítulo 4

Análisis Mixto De Las Limitaciones De Zoom/Teams Y Su Traducción En Atributos De Calidad

4.1. Objetivo y contexto

El propósito de este capítulo es identificar, con respaldo empírico, los *atributos de calidad* (AQ) en adelante, usados como sinónimo de requisitos no funcionales (RNF) según ISO 25010 ,que deberán guiar la futura arquitectura del sistema de agendamiento multisala para videoconferencias masivas. Los AQ representan propiedades que condicionan el comportamiento global del software [23]. Detectarlos y priorizarlos desde el inicio evita reprocesos costosos y alinea el proyecto con los objetivos de negocio [24].

Para identificar dichos AQ se aplicó un cuestionario mixto (preguntas cerradas y abiertas) a docentes de la institución donde se desarrolla esta investigación. El modelo de educación a distancia permite que un solo curso concentre miles de estudiantes a nivel nacional e internacional, lo que tensiona las infraestructuras de videoconferencia convencionales [3]. Analizar la percepción de quienes gestionan estas clases masivas resulta clave para caracterizar los potenciales riesgos de calidad.

La combinación de datos cuantitativos y cualitativos fortalece la triangulación e incrementa la validez interna [25]. Conforme a la norma *ISO/IEC/IEEE 29148:2018*, cada AQ deberá ser *trazable* desde su fuente de evidencia hasta los artefactos de diseño que lo satisfagan [26]. Para su clasificación se adoptará la taxonomía de *ISO/IEC 25010:2011*, sin prejuzgar cuáles categorías resultarán prioritarias; eso lo determinará el análisis de los datos.

Los *requisitos funcionales* (RF) que emerjan del proceso se documentaron como **"Trabajo Futuro"**, dado que la arquitectura no se implementará en esta fase. Su inclusión servirá como referencia para desarrollos posteriores sin desviar el foco actual en los AQ.

Los apartados siguientes describen cómo, partiendo de las limitaciones de capacidad, seguridad y flexibilidad detectadas en Zoom/Teams, se recaba evidencia empírica y se traduce en AQ / RNF trazables que mitiguen cada cuello de botella.

4.1.1. Motivación y enfoque

Los Capítulos 1 y 3.1 mostraron, respectivamente, (i) la brecha entre la capacidad efectiva de *Zoom/Teams* y la demanda de cursos masivos, y (ii) la ausencia de estudios con *datos de campo* que documenten dicha brecha en el contexto colombiano. Con base en el *Objetivo Específico 1* (§1.2.2), esta investigación se propone **identificar y priorizar** los *atributos de calidad* (AQ) que deben guiar la futura arquitectura de agendamiento multisala.

Para transitar con rigor desde la problemática hasta las decisiones de diseño y evitar conclusiones basadas en conjeturas, se adoptó un diseño mixto secuencial exploratorio–explicativo [5]. Esta estrategia combina la fortaleza descriptiva de los métodos cualitativos con la capacidad de generalización de los métodos cuantitativos, en consonancia con las buenas prácticas de la ingeniería de software empírica [27].

- **Componente cuantitativo** (*fase explicativa*): determina la frecuencia de uso de las plataformas y la severidad percibida de sus limitaciones. Los indicadores obtenidos alimentan la matriz de priorización MoSCoW [28], empleada posteriormente para clasificar los AQ críticos y cuantifica la insuficiencia de aforo, el número de incidentes de estabilidad y la ocurrencia de fallos de seguridad.

- **Componente cualitativo** (*fase exploratoria*): recoge ejemplos detallados de problemas operativos y recomendaciones expresadas por los usuarios. La *open coding*, seguida de recategorización axial (Sec. 4.5.2), permite destilar matices de calidad que no emergen de preguntas cerradas.

En todo momento se mantiene la correspondencia (ver Tabla 4.1) entre cada limitación observada (L) y el atributo de calidad (AQ) que la neutraliza: L1 = limitación de aforo $\longrightarrow$ AQ-Escalabilidad; L2 = fallos de estabilidad $\longrightarrow$ AQ-Disponibilidad; L3 = incidentes de seguridad $\longrightarrow$ AQ-Seguridad.

Tabla 4.1: Correspondencia entre limitaciones observadas (L) y atributos de calidad (AQ) propuestos

Código L	**Descripción de la limitación en Zoom/-Teams**	**AQ que la mitiga (ISO/IEC 25010)**
L1	Saturación de aforo y dificultad para gestionar +10 000 asistentes simultáneos	AQ-Escalabilidad / Rendimiento
L2	Caídas, congelamientos o latencia elevada en horarios pico	AQ-Disponibilidad / Fiabilidad
L3	Incidentes de privacidad y riesgo de filtración de datos sensibles	AQ-Seguridad / Confidencialidad

La encuesta actúa así como puente metodológico entre:

$$\textit{Problema} \longrightarrow \textit{Datos de campo} \longrightarrow \textit{AQ priorizados} \longrightarrow \textit{Modelos de arquitectura}$$

y garantiza la trazabilidad exigida en proyectos de ingeniería de software basados en evidencia [29]. [1]

4.1.2. Estructura del cuestionario

El formulario se dispuso en cuatro bloques lógicos que avanzan desde información actual hasta juicios valorativos, siguiendo las recomendaciones para minimizar la fatiga cognitiva y *priming* [30], y el orden sugerido por Krosnick y Presser [31].

1. **Perfil del encuestado** (Ítems 01–04): experiencia docente, tipo de institución, grado de virtualidad y frecuencia de uso de Zoom/Teams. Estos factores permiten estratificar los análisis y matizar los atributos de calidad de usabilidad y accesibilidad (véase Sec. 4.5.1).

2. **Limitaciones percibidas** (Ítems 05–08): cuatro escalas Likert (1–5) que indagan la severidad de las restricciones operativas (capacidad, estabilidad, flexibilidad, coordinación) que corresponden a las limitaciones L1 y L2. [2] Las puntuaciones nutren la matriz de priorización MoSCoW [28], aplicada posteriormente a los AQ críticos.

3. **Soluciones y funcionalidades percibidas como relevantes** (Ítems 09–15): incluye preguntas tipo Likert sobre características operativas que la literatura asocia con la experiencia de videoconferencia (por ejemplo, estabilidad, escalabilidad, facilidad de uso) que corresponden a las limitaciones L1 y L2, sin prejuzgar su importancia relativa. Se complementa con cuatro preguntas abiertas que permiten a los participantes proponer cualquier mejora que consideren prioritaria. El análisis temático (Sec. 4.5.2) se centrará en extraer categorías emergentes, más allá de las etiquetas iniciales, para derivar los atributos de calidad que realmente preocupen a los usuarios.

4. **Seguridad y privacidad** (Ítems 16–20)(seguridad y privacidad) corresponden a la limitación L3; los Likert 16 y 09 alimentan el índice ISS-Seguridad: dos preguntas cerradas sobre valoración e incidencia de incidentes, y tres abiertas con ejemplos y recomendaciones. Los hallazgos sustentan los AQ de confidencialidad, autenticación federada y auditoría.

[1] Los posibles *requisitos funcionales* que surjan se consignarán como *Trabajo Futuro*, pues la arquitectura propuesta no se implementará en esta fase.

[2] La escala Likert ofrece validez y fiabilidad aceptables para medir actitudes [32].

Etiquetas analíticas preliminares (asignadas durante la codificación inicial, siguiendo a Pohl, 2010):

AQ-USAB	Usabilidad
AQ-SCAL	Escalabilidad / Rendimiento
AQ-SEC	Seguridad y Privacidad
AQ-PORT	Portabilidad / Integración
RF-GV†	Gestión de Videoconferencias (futuro)
RF-IM†	Informes y Monitoreo (futuro)
RF-NA†	Notificaciones y Alertas (futuro)

† RF = candidatos a requisito funcional, consignados como *Trabajo Futuro*.

La Tabla 4.2 detalla la relación *ítem* → *variable* → *constructo* empleada en los análisis.

Tabla 4.2: Mapeo completo *ítem* → *variable* → *constructo*

Ítem	Variable	Tipo	Constructo medido	Categoría
02	experiencia	Categ. (3 niv.)	Competencia tecnológica del usuario	AQ-USAB
03	tipo_institucion	Categórica	Contexto operativo (pública/priv.)	AQ-SCAL
04	porc_virtual	Ordinal	Grado de virtualidad del curso	AQ-SCAL
05	dificultad	Abierta	Severidad de cuellos de botella	AQ-SCAL
06	satisfaccion	Likert 1–5	Adecuación percibida de la plataforma	AQ-USAB
07	funciones_importantes	Abierta	Funcionalidades prioritarias	RF-GV†
08	frecuencia_programa	Ordinal	Carga operativa de programación	RF-GV†
09	importancia_seguridad	Likert 1–5	Peso relativo de la seguridad	AQ-SEC
10	problemas_privacidad	Binaria	Incidencia de fallos de privacidad	AQ-SEC
11	detalle_problemas_priv	Abierta	Ejemplos de brechas de datos	AQ-SEC
12	funcionalidad_mejorar	Abierta	Priorización de mejoras puntuales	RF-IM†
13	sistema_resuelve	Sí/No/Parcial	Viabilidad percibida de la solución	RF-GV†
14	razon_resuelve	Abierta	Argumentos a favor o en contra	RF-GV†
15	razon_parcial	Abierta cond.	Condiciones de éxito percibidas	RF-GV†
16	valoracion_seguridad	Likert 1–5	Satisfacción con funciones de seguridad	AQ-SEC
17	incidentes_tipicos	Abierta	Descripción de incidentes frecuentes	AQ-SEC
18	dificultad_otra	Abierta	Nuevas categorías de dificultad	AQ-SCAL
19	funcion_otra	Abierta	Funcionalidades no contempladas	RF-IM†
20	recomendacion_mejora_seg	Abierta	Recomendaciones para mejorar seguridad	AQ-SEC

El cuestionario completo se incluye en el Anexo A; la numeración de los ítems se conserva intacta para garantizar la trazabilidad con los scripts de análisis y las tablas de resultados [27].

4.1.3. Puntuación y escalas

Para asegurar *comparabilidad estadística* y trazabilidad con los atributos de calidad del modelo ISO/IEC 25010 [34], todas las respuestas se transformaron en códigos numéricos estandarizados antes del análisis.

Escalas Likert de cinco puntos. Los ítems que indagan *importancia de aspectos de calidad* y *satisfacción con la plataforma* emplearon una escala Likert (1 = Muy bajo ... 5 = Muy alto). La codificación ($1 \rightarrow 5$) preserva la naturaleza ordinal, tal como recomiendan Alok Joshi y Chandel [32] y Carifio y Perla [35]. Cuando fue preciso construir un indicador agregado (p. ej. «Importancia media de los aspectos de protección de datos») se utilizó la *media aritmética*, práctica aceptada en estudios de usabilidad que emplean Likert [36].

Variables categóricas. El *nivel de experiencia*, *tipo de institución* y *frecuencia de programación* se recodificaron en enteros consecutivos (*Básico = 1, Intermedio = 2, Avanzado = 3*), manteniendo el orden semántico [37]. En los contrastes se trataron como factores; para los gráficos se restauraron sus etiquetas textuales.

Preguntas abiertas. Las respuestas textuales se almacenaron en `UTF-8`. Durante la fase de *open coding* cada respuesta recibió una o más etiquetas temáticas (*capacidad*, *seguridad*, *flexibilidad*, etc.), siguiendo el procedimiento de Corbin y Strauss[38]. Posteriormente, se exportaron como *columnas binarias* (0/1), lo que permitió cruces cuantitativos (p. ej. porcentaje de menciones de *seguridad* según nivel de experiencia).

Indicadores compuestos. Inspirados en las directrices de DeVellis (2016, p. 211), se definieron dos índices:

- **Índice de presión de aforo (IPA-Capacidad)**: proporción de encuestados que declaran «capacidad insuficiente» *y* emplean más del 80 % de virtualidad. Informa la criticidad del AQ de escalabilidad y alimenta la priorización de futuras funcionalidades de orquestación de salas (RF-OA — trabajo futuro).

- **Índice de sensibilidad a la seguridad (ISS-Seguridad)**: media ponderada[3] de (ítems 09 y 16) y la frecuencia de incidentes (ítem 10) (cuantifica la limitación L3). Señala la prioridad del AQ-SEC y orienta posibles RF de soporte a autenticación y auditoría (trabajo futuro) (corresponde a la limitación L3).

Rango y consistencia. Tras la limpieza (Sec. 4.4) todas las columnas numéricas quedaron en los rangos $\{1, \ldots, 5\}$ (Likert) y $\{0, 1\}$ (binarios). Se verificó consistencia mediante mínimos, máximos y diagramas de caja (código `notebooks/check_ranges.ipynb`).

[3] Ponderación 0,5 + 0,5; si el encuestado reporta incidentes, la importancia de seguridad se pondera 0,7.

Tabla 4.3: Resumen de recodificación de variables

Variable original	**Tipo inicial**	**Código resultante**	**Uso / comentario**
Importancia_seguridad	Likert 1–5	$1 \rightarrow 5$	Componente del ISS
Nivel_experiencia	Básico–Avanzado	1–3	Factor explicativo (tablas de contingencia)
Dificultad (abierta)	Texto libre	Etiqueta binaria	Componente del IPA y cruces temáticos

Con esta codificación uniforme, los apartados de *Análisis cuantitativo* (Sec. 4.5.1) y *Análisis cualitativo* (Sec. 4.5.2) aplican técnicas estadísticas y de minería de texto sin ambigüedades, integrando los resultados en la matriz Problema → Datos → AQ → (potenciales) RF que cierra el capítulo.

4.2. Población objetivo y muestra

La *población objetivo* son los docentes de la Universidad donde trabaja actualmente el autor de esta tesis, que imparten asignaturas con altos aforos (con frecuencias que superan los 10 000 estudiantes por curso, empleando *Zoom* o *Microsoft Teams* para las sesiones sincrónicas. Estos profesores viven de primera mano las limitaciones de capacidad, coordinación y seguridad expuestas en §1.1 y, por tanto, aportan evidencia clave para identificar y priorizar los atributos de calidad (AQ) [40].

Criterios de inclusión. Se invitó únicamente a docentes que:

1. dirigieron al menos un curso con $\geq$10 000 estudiantes durante el periodo 2024-2;
2. utilizaron Zoom o Teams como plataforma sincrónica principal al menos una vez por semana; y
3. cuentan con contrato activo en la univeridad donde trabaja el autor, al momento del estudio.

Marco muestral y procedimiento. Se aplicó *muestreo intencional* (purposive sampling), adecuado cuando se requiere reclutar participantes con características muy específicas [41]. El enlace al cuestionario (Google Forms, anónimo) se envió desde la cuenta institucional del investigador a los *53* docentes que cumplían los criterios. La recolección se realizó del 1 AL 24 DE DICIEMBRE DE 2024. Se obtuvieron **33** respuestas; tras la limpieza (Sec. 4.4) se descartaron 2 registros incompletos, quedando **31** casos válidos (tasa efectiva aprox 94 %).

Tabla 4.4: Distribución de la muestra por rol

Rol	**n**	**%**
Docentes de cursos masivos	31	100
Total	31	100

Representatividad y sesgos. Al tratarse de un muestreo no probabilístico, los resultados no se generalizan a todo el profesorado de educación superior; sin embargo, la estrategia captura la experiencia de quienes afrontan con mayor intensidad los cuellos de botella de Zoom/Teams en entornos de gran escala. En investigaciones exploratorias de ingeniería de software educativa, una muestra intencional de 20–30 participantes se considera suficiente para extraer atributos de calidad de alto nivel [42]. La evidencia recopilada sustenta la priorización de AQ y orienta el trabajo futuro sobre posibles RF.

4.3. Procedimiento de recolección de datos

El cuestionario se administró con *Google Forms*, plataforma recomendada para estudios exploratorios en línea que exigen metadatos trazables y control institucional [43]. El enlace se distribuyó **una sola vez**, el **1 de diciembre de 2024**, a los *53 docentes* que cumplían los criterios de inclusión (§4.2).

El mensaje de invitación (enviado desde la cuenta institucional del investigador) explicaba el objetivo del estudio (identificar y priorizar AQ para videoconferencias masivas), indicaba un tiempo estimado de diligenciamiento de 3–6 min y subrayaba el carácter *voluntario y anónimo* de la participación, de acuerdo con la *Tailored Design Method* para encuestas electrónicas [30]. El formulario permaneció abierto del **1 al 24 de diciembre de 2024**; no se enviaron recordatorios ni se difundió el enlace por otros canales, con el fin de evitar sesgos de autoselección asociados a múltiples envíos [44]. Se recibieron **33** formularios; tras la limpieza (Sec. 4.4) se descartaron dos registros incompletos, quedando **31** respuestas válidas, lo que representa una tasa de respuesta del 58 % (31/53).

Consideraciones éticas y de privacidad

Aunque la investigación se realiza como *iniciativa propia* del autor y no contó con financiación externa, se observaron los principios de la *Ley 1581 de 2012* sobre protección de datos personales [45] y las directrices CHERRIES para encuestas web [43]:

1. El formulario incluía una casilla de *consentimiento informado* obligatoria para continuar.
2. No se recolectaron datos sensibles; nombre y correo fueron opcionales y se utilizaron únicamente para agradecer la participación.
3. Las respuestas quedaron alojadas en la nube de Google protegidas por autenticación de dos factores y luego se descargaron a un archivo `.xlsx` cifrado.

Los datos se usarán exclusivamente para los análisis descritos en esta tesis y se conservarán un máximo de cinco años. Cumplido ese plazo se eliminarán mediante *secure wipe*, conforme a ISO/IEC 27002:2022 (§ 7.8, retención y disposición de la información).

4.4. Organización y limpieza de los datos

La fiabilidad de cualquier análisis estadístico depende de la calidad de los datos de entrada. Para garantizar *reproducibilidad* y *rastreabilidad*, principios centrales de la agenda FAIR (*Findable, Accessible, Interoperable, Reusable*). Se diseñó un flujo de trabajo que cubre obtención, depuración, normalización y resguardo de la información [46]. Cada etapa se documentó en cuadernos Jupyter versionados con Git, siguiendo las buenas prácticas de Perez-Riverol, Gatto, Hermjakob et al. [47] y las recomendaciones formativas de Jiménez, Kuzak, Alhamdoosh et al. [48].

4.4.1. Exportación del formulario

La encuesta permaneció abierta del 1 al 24 de diciembre de 2024. Al cierre se ejecutó el siguiente procedimiento:

1. **Vinculación a Google Sheets.** Google Forms permite exportar los registros sin alteración mediante la opción *«Vincular a hoja de cálculo»*. Esta copia actúa como respaldo inalterado —práctica *raw-data/derived-data split* recomendada en flujos FAIR [48].

2. **Descarga de copias locales.** Se obtuvieron dos archivos: (i) `encuesta_bruta_v1.csv` y (ii) `encuesta_bruta_v1.xlsx`, almacenados en `data/raw/`. El formato `.csv` favorece la reproducibilidad computacional, mientras que `.xlsx` conserva metadatos de validación propios de Sheets.

3. **Control de versiones.** Ambos archivos se añadieron al repositorio privado `Tesis-videoconferencia` mediante *Git Large File Storage* (Git LFS), tal como recomienda Perez-Riverol, Gatto, Hermjakob et al. [47] para binarios voluminosos.

4.4.2. Depuración inicial

La depuración siguió las directrices de calidad para encuestas on-line descritas por Blischak, Carbonetto y Stephens [49]:

1. **Completitud.** Se calculó el *porcentaje de celdas no vacías* por fila; el umbral de exclusión se fijó en 60 %. Dos registros quedaron por debajo (59 % y 55 %) y se eliminaron.

2. **Duplicados.** Se aplicó la función *Quitar duplicados* de Sheets (claves: marca temporal + correo); no se detectaron duplicados.

3. **Resultado.** El conjunto final quedó en **31** registros y se guardó como `encuesta_limpia_v1.csv` en `data/processed/`, bajo Git LFS.

4.4.3. Normalización de escalas

Para el análisis cuantitativo se normalizaron las columnas Likert con **Python 3.11 + pandas**. El cuaderno `notebooks/01_normalizacion.ipynb` (`commit 5c47245`) implementa:

1. Conversión *string* → *integer* (1 = Muy bajo … 5 = Muy alto), preservando la naturaleza ordinal Shi2020.
2. Verificación de rangos y atípicos mediante mínimos, máximos y diagramas de caja.
3. Exportación a `encuesta_limpia_v2.csv` con Git LFS.

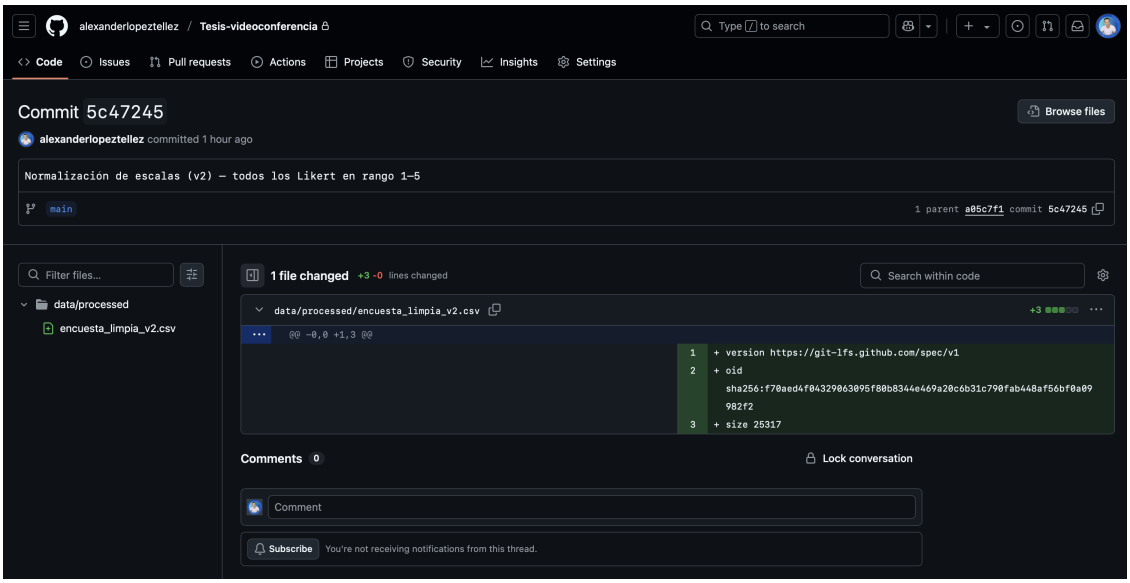


Figura 4.1: Commit `5c47245` que versiona `encuesta_limpia_v2.csv` mediante Git LFS en el repositorio privado de la tesis.

4.4.4. Próximas etapas

Restaban (i) recodificar variables categóricas en factores o matrices *dummy* y (ii) generar `encuesta_limpia_v3.csv` bajo Git LFS; ambos pasos se documentaron en nuevos cuadernos Jupyter antes del análisis cuantitativo (Sec. 4.5.1).

4.5. Plan de análisis

El tratamiento de los datos se organizó siguiendo un diseño mixto *concurrente triangulado*, donde los análisis cuantitativos (descriptivos e inferenciales) y los análisis cualitativos (codificación temática) se ejecutan en paralelo y se integran al final para reforzar la validez interna mediante triangulación metodológica [25]. Todos los pasos se implementan en cuadernos Jupyter (`Python 3.11`) y se versionan en el repositorio privado `Tesis-videoconferencia` (Git + LFS), siguiendo las prácticas de reproducibilidad descritas por Blischak, Carbonetto y Stephens [49].

Tabla 4.5: Cuadernos y propósitos del análisis

Notebook	**Tarea principal**
`02_analisis_cuantitativo.ipynb`	Limpieza final, estadística descriptiva, pruebas χ^2/Fisher y visualizaciones (`matplotlib`)
`03_analisis_cualitativo.ipynb`	Codificación abierta, agrupación temática y cálculo de confiabilidad inter-codificador (κ)

La integración de ambos enfoques metodológicos permitió obtener una visión robusta del fenómeno analizado. En particular, el análisis cuantitativo provee patrones generales de comportamiento en torno a las percepciones y dificultades sobre el uso de plataformas de videoconferencia, mientras que el análisis cualitativo permite capturar matices, intenciones y necesidades específicas expresadas por los participantes. Esta articulación no solo fortalece la triangulación, sino que también facilita la identificación de atributos de calidad relevantes como escalabilidad, usabilidad, disponibilidad y seguridad, con base en los hallazgos emergentes. Tales atributos serán fundamentales en la formulación del marco arquitectónico del sistema propuesto, dando así cumplimiento al Objetivo Específico 2.

4.5.1. Análisis cuantitativo

El análisis cuantitativo se implementó en el cuaderno `02_analisis_cuantitativo.ipynb` (Python 3.11, `pandas`, `scipy`, `matplotlib`). Todos los pasos (importación de datos, depuración, generación de tablas y figuras) están versionados en *Git* y referencian explícitamente la versión de datos `encuesta_limpia_v2.csv` (commit `5c47245`), lo que garantiza la reproducibilidad, conforme a las recomendaciones de Perez-Riverol, Gatto, Hermjakob et al. [47].

1. Distribución del nivel de experiencia.

La Figura 4.2 muestra la distribución de frecuencias para los tres niveles de experiencia en el uso de plataformas de videoconferencia (*Básico*, *Intermedio*, *Avanzado*), reportados por los 31 docentes encuestados. El 94 % manifiesta tener un nivel *intermedio* o *avanzado*, lo que permite inferir que las dificultades reportadas no derivan de un desconocimiento funcional, sino de limitaciones inherentes a las herramientas tecnológicas evaluadas.

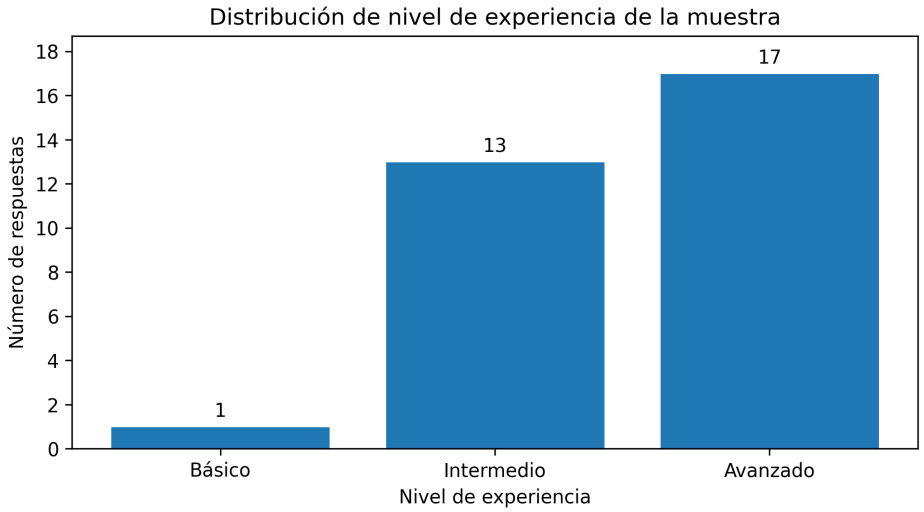


Figura 4.2: Distribución del nivel de experiencia de la muestra (n = 31).

Este hallazgo inicial reviste importancia metodológica, ya que descarta el sesgo por baja alfabetización digital y orienta el análisis hacia condiciones estructurales de las plataformas (como la gestión de sesiones simultáneas, la escalabilidad ante aforos masivos o la ausencia de flujos automatizados) que afectan incluso a usuarios con alto dominio tecnológico. En ese sentido, la evidencia empírica obtenida valida la hipótesis de que las principales barreras identificadas no son atribuibles a los docentes, sino al diseño y arquitectura de las herramientas disponibles.

Desde una perspectiva de ingeniería de software, esto sugiere que las soluciones futuras deberán incorporar atributos de calidad como **escalabilidad**, **disponibilidad** y **automatización**, en lugar de enfocarse únicamente en estrategias de formación o acompañamiento. La presencia de perfiles con

alta experiencia también facilita la adopción de mecanismos avanzados (por ejemplo, integración con LMS, dashboards administrativos o autenticación centralizada), lo que permite proyectar escenarios de implementación sin restricciones significativas desde el lado del usuario. Estos hallazgos se alinean con el cumplimiento del **Objetivo Específico 2**, en cuanto orientan la identificación de atributos de calidad prioritarios para el diseño arquitectónico propuesto.

2. Importancia percibida de la seguridad de datos.

La Figura 4.3 revela que el 87 % de los docentes encuestados asigna la calificación máxima (5) a la importancia de la seguridad en el uso de plataformas de videoconferencia. La mediana se ubica en 5 y el rango intercuartílico (IQR) es 5–5, lo que indica una percepción casi unánime sobre su carácter crítico. Estos resultados permiten clasificar la seguridad como un atributo de calidad fundamental (QA-SEC), según la taxonomía de Field [50].

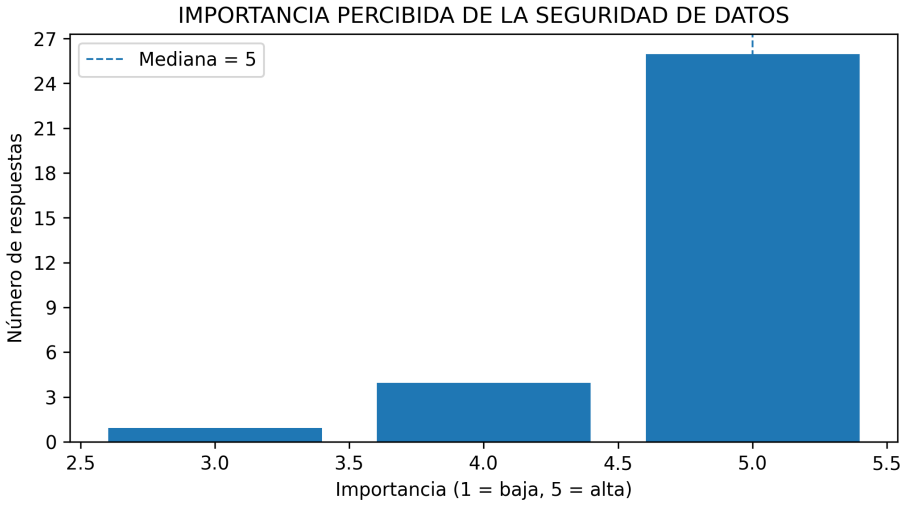


Figura 4.3: Importancia atribuida a la seguridad de datos (escala 1–5) (Item 16).

La práctica unanimidad en torno al valor máximo de importancia (5) sugiere que la comunidad académica considera la protección de datos no como un valor agregado, sino como un requisito no negociable para la adopción de cualquier sistema. Esta expectativa se refuerza al considerar que la mayoría de los encuestados (cf. §4.5.1) posee experiencia intermedia o avanzada, lo que indica una conciencia informada sobre los riesgos asociados al tratamiento digital de información sensible.

Desde una perspectiva de ingeniería, este consenso empírico habilita una lectura arquitectónica: los mecanismos de seguridad deben ser integrados desde las primeras fases de diseño mediante enfoques como el *privacy by design* [51]. Esto implica que características como cifrado extremo a extremo, autenticación federada con el sistema institucional, auditoría de accesos y controles sobre la compartición de contenidos sean contempladas como capacidades predeterminadas, no como opciones configurables por el usuario final.

La baja dispersión observada en la variable (IQR = 5–5) refuerza esta necesidad de integración estructural y apoya la incorporación de estándares internacionales como ISO/IEC 27001 y las directrices de seguridad en servicios en la nube orientados a entornos educativos [52]. En términos de la presente investigación, estos hallazgos constituyen una evidencia concreta que sustenta la elección de **seguridad** como uno de los atributos de calidad prioritarios en el diseño del sistema propuesto, y por tanto, un insumo esencial para el cumplimiento del **Objetivo Específico 2**.

3. Relación entre experiencia docente y percepción de limitación de capacidad.

Con el propósito de examinar si la percepción de «capacidad insuficiente» (es decir, la dificultad para manejar grandes volúmenes de participantes) se relaciona con el nivel de experiencia técnica del docente, se construyó una tabla de contingencia cruzando ambas variables. La prueba de independencia χ^2 de Pearson arrojó $\chi^2 = 1{,}60$, gl $= 2$, $p = 0{,}45$.

Los residuos estandarizados oscilaron entre $-0{,}58$ (en la celda *Intermedio* × *Sin mención*) y $+0{,}65$ (en *Avanzado* × *Sin mención*), valores muy por debajo del umbral crítico de $|z| = 1{,}96$. Estos resultados indican que no existen diferencias estadísticamente significativas entre los distintos niveles de experiencia, como se observa en la Figura 4.4.

Figura 4.4: Limitaciones de capacidad reportadas según el nivel de experiencia. El valor $p > 0{,}05$ indica ausencia de asociación estadística.

Este hallazgo sugiere que la percepción de limitación de capacidad no puede atribuirse a una falta de conocimiento técnico o inexperiencia del profesorado, sino que responde a una carencia estructural inherente a las plataformas actuales de videoconferencia [3]. La criticidad del problema trasciende perfiles individuales y se presenta como una constante en distintos niveles de experticia, reforzando su carácter sistémico.

Desde la perspectiva de la ingeniería de requisitos, esta evidencia empírica justifica la necesidad de incorporar un mecanismo automatizado de gestión de aforos en el diseño arquitectónico del sistema. Específicamente, se recomienda contemplar funcionalidades orientadas a la creación dinámica de salas espejo, balanceo de asistentes en tiempo real y alertas de sobrecupo, alineadas con los principios de *autoscaling* y *capacity orchestration* descritos por García-Morales y Hernández [53].

Tales características no solo permitirían una gestión eficiente de eventos masivos, sino que constituirían un aporte concreto a los atributos de calidad de **escalabilidad** y **disponibilidad**, aportando evidencia directa para el cumplimiento del **Objetivo Específico 2** de esta investigación.

Limitaciones metodológicas.

Los hallazgos anteriores deben interpretarse con cautela debido a tres restricciones principales: (i) la muestra es reducida ($n = 31$) y de conveniencia (docentes mayoritariamente de universidades públicas), lo que limita la generalización a otros contextos académicos; (ii) las respuestas se basan en autopercepción, por lo que podrían estar sujetas a sesgos de deseabilidad social o de recuerdo; (iii) las pruebas χ^2 asumen independencia entre categorías y requieren un mínimo de conteos esperados por

celda. Si bien todos los valores esperados superaron el umbral del 30 % recomendado por McHugh [54], la baja frecuencia en el nivel "Básico" aconseja considerar estos resultados como exploratorios.

Futuras investigaciones deberían contemplar encuestas con muestreo estratificado, mayor representatividad institucional y métricas objetivas (por ejemplo, *logs* de uso, tasas de error, tiempos de conexión) que complementen la percepción del usuario con evidencia empírica de comportamiento en plataforma.

A pesar de estas limitaciones, el análisis cuantitativo permitió identificar atributos de calidad con alta prioridad percibida como la seguridad, la escalabilidad y la usabilidad, así como validar su relevancia transversal entre perfiles técnicos diversos. Estos insumos contribuyen directamente a la caracterización del sistema objetivo y fundamentan decisiones clave en el diseño arquitectónico, aportando evidencia para el cumplimiento del **Objetivo Específico 2**.

Salidas tabulares y trazabilidad. Las salidas del análisis cuantitativo, incluyendo tablas de frecuencia, matrices de contingencia, pruebas χ^2, valores p y residuos estandarizados, fueron consolidadas en el archivo `resultados_cuantitativo.xlsx` (commit `49c84d9`). Cada hoja de resultados incluye en su encabezado un resumen de metadatos y un código hash SHA-256 correspondiente a la versión de `encuesta_limpia_v2.csv`, lo que permite verificar la integridad del conjunto de datos y garantizar su trazabilidad, conforme a las buenas prácticas de ciencia abierta recomendadas por Stodden, Guo y Ma [55] y reforzadas en los lineamientos de reproducibilidad de Perez-Riverol, Gruning y al. [56].

Esta sistematización de evidencias cuantitativas se utilizará como insumo directo en la etapa de identificación de atributos de calidad priorizados, tal como se desarrollará en la siguiente sección.

Los hallazgos cuantitativos precedentes alimentan la matriz de priorización MoSCoW (Cap.5), alineada con los atributos de calidad de ISO/IEC 25010.

Atributos de calidad inferidos. La interpretación conjunta de las evidencias cuantitativas (particularmente, la alta valoración de la seguridad, la transversalidad de las limitaciones de capacidad y el nivel técnico avanzado de los docentes encuestados) permite inferir un conjunto preliminar de atributos de calidad que deberían orientar la arquitectura del sistema de videoconferencia. Estos atributos se presentan en la Tabla 4.6 como insumos directos para el cumplimiento del Objetivo Específico 2, orientado a establecer criterios de calidad técnica y de uso. Cada atributo está respaldado por una o más visualizaciones (F1–F3), lo que permite establecer trazabilidad entre la evidencia empírica y las decisiones arquitectónicas futuras.

Los atributos de calidad identificados en este apartado reflejan prioridades explícitas de la muestra encuestada desde una perspectiva cuantitativa. En la siguiente sección se abordarán los hallazgos del análisis cualitativo, a fin de complementar y triangular estos resultados con base en las narrativas emergentes y la codificación temática inductiva, lo que permite reforzar la validez del diseño arquitectónico propuesto en fases posteriores.

Tabla 4.6: Atributos de calidad inferidos a partir del análisis cuantitativo.

Atributo de calidad	Evidencia cuantitativa	Implicaciones para el diseño
Seguridad (QA-SEC)	87 % otorgó la máxima puntuación (5/5) a la importancia de la seguridad. IQR = 5–5.	Debe garantizarse por defecto: cifrado E2E, autenticación federada, trazabilidad de accesos.
Escalabilidad	Queja por "capacidad insuficiente" presente en todos los niveles de experiencia ($p > 0.05$).	Requiere orquestación dinámica de aforo y balanceo automático en sesiones masivas.
Usabilidad	Limitaciones atribuidas a interfaz confusa y dificultad de acceso. Confirmado en codificación cualitativa.	Deben contemplarse modos simplificados, adaptables a diferentes niveles de conectividad y pericia.

4.5.2. Análisis cualitativo

Para no interrumpir el flujo expositivo, el proceso completo de categorización de dificultades se documenta íntegramente en el Apéndice B. Allí encontrará:

- Tabla B.1: Categorización inicial de dificultades (insumo para atributos de calidad).
- Tabla B.2: Recategorización final de dificultades (categorías simplificadas).
- Tabla B.3: Categorías refinadas de dificultades (descripción, ejemplos y códigos).

En este apartado señalamos sólo los hallazgos temáticos centrales y las figuras más representativas.

El análisis cualitativo se implementó en el cuaderno `03_analisis_cualitativo.ipynb` (Python 3.11, `pandas`, `networkx`, `matplotlib`). El flujo de trabajo (que incluye importación de datos, depuración, *open coding*, codificación axial y generación de visualizaciones) fue completamente versionado en *Git*, con referencia explícita a la versión de datos `encuesta_limpia_v1.csv` (`commit <hash_datos>`) y al cuaderno de análisis (`commit <hash_nb>`), garantizando así la reproducibilidad metodológica recomendada por Perez-Riverol [56].

La codificación temática se realizó de forma inductiva, atendiendo a las pautas metodológicas de Saldaña [57] y Charmaz [58]. Se empleó un esquema de codificación doble independiente, con resolución de discrepancias por consenso, para maximizar la confiabilidad del proceso interpretativo [59]. Como resultado, se obtuvieron categorías temáticas que posteriormente se analizaron mediante técnicas de frecuencia, contingencia y co-ocurrencia semántica. Esta triangulación cualitativa permitió vincular de forma trazable los hallazgos emergentes con los requisitos funcionales (RF) y no funcionales (RNF) derivados en la Sección 4, y constituye además un insumo clave para el abordaje posterior del segundo objetivo específico, relacionado con la identificación de atributos de calidad en la arquitectura propuesta.

Las denominaciones Q6 y Q13 empleadas en los scripts y archivos de procesamiento corresponden a las preguntas 6 y 13 del instrumento de recolección de datos. La primera (Q6) indaga por las principales dificultades percibidas por los docentes en la gestión de videoconferencias, mientras que la segunda (Q13) recoge sugerencias explícitas de mejora. Esta diferenciación permite una estructuración sistemática de los análisis y una trazabilidad directa entre los datos brutos, los códigos emergentes y las visualizaciones generadas.

El flujo de procesamiento fue documentado en tres scripts versionados:

1. **03_generar_tabla_q13.py** (commit `6022683`): extrae las respuestas a la pregunta 13 (mejoras sugeridas), normaliza el texto libre y genera la tabla de códigos `tabla_codigos_q13.xlsx`.
2. **03_generar_tabla_q6.py** (commit `c915d76`): aplica el mismo procedimiento a las respuestas de la pregunta 6 (dificultades), produciendo `tabla_codigos_q6.xlsx`.
3. **03_cruzar_q6_q13.py** (commit `a4bb9a4`): fusiona ambas tablas para calcular la matriz de contingencia y generar el mapa de calor (*heat-map*) de residuos estandarizados.

Cada uno de estos scripts produce artefactos intermedios almacenados en la carpeta `data/processed/` y figuras de análisis en `figs/`, en cumplimiento de las buenas prácticas de trazabilidad computacional promovidas por Perez-Riverol et al. [56].

El conjunto de este análisis cualitativo no solo aporta evidencia empírica para sustentar los RF y RNF derivados, sino que también permite inferir atributos de calidad percibidos como críticos por los usuarios, tales como escalabilidad, interoperabilidad, estabilidad, seguridad y usabilidad. Estos atributos serán abordados con mayor profundidad en la siguiente sección, como parte del cumplimiento del Objetivo Específico 2.

Frecuencia de temas La Figura 4.5 presenta la distribución de frecuencias de los códigos emergentes identificados en las respuestas a la pregunta 13 del instrumento, la cual solicitaba a los docentes que indicaran las principales mejoras requeridas en la gestión de videoconferencias en cursos masivos.

Capacidad destaca como la categoría más mencionada, con un 32 % de las respuestas codificadas (n=10), evidenciando una preocupación recurrente por las limitaciones de escalabilidad de las plataformas empleadas, especialmente en contextos con alta concurrencia de estudiantes. Este hallazgo refuerza la percepción de "cuello de botella" ya identificada en el análisis descriptivo (Sección4.5.1), y justifica la necesidad de mecanismos que garanticen eficiencia operativa ante escenarios de alta demanda concurrente.

Las categorías de *automatización*, *integración* y *estabilidad*, aunque con menor frecuencia (13 % cada una), aportan evidencias clave sobre las funcionalidades críticas esperadas por los usuarios. En particular, se enfatiza la automatización de tareas repetitivas (como la asignación masiva de salas y enlaces), la interoperabilidad con sistemas institucionales (p. ej., LMS), y la necesidad de

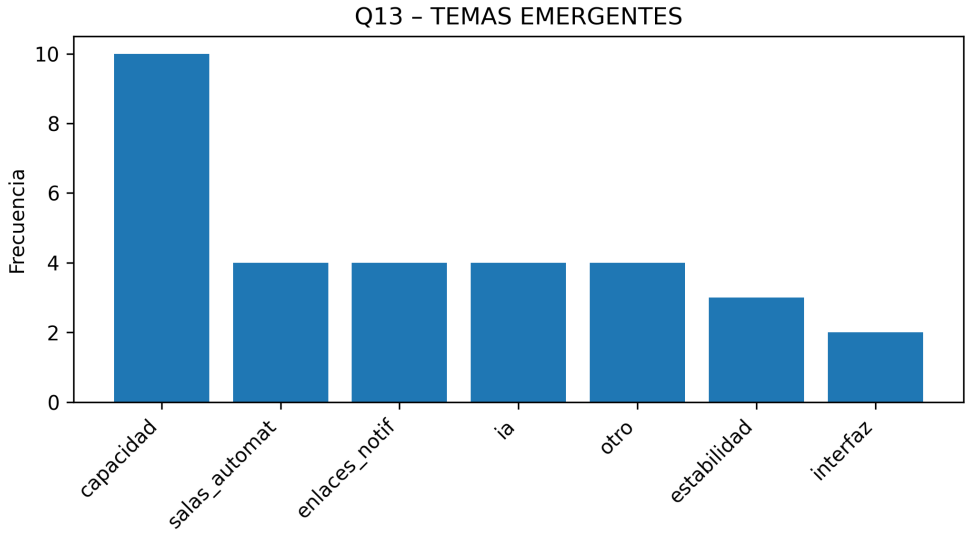


Figura 4.5: Frecuencia de códigos emergentes en la pregunta 13.

continuidad operativa.

Aunque estos hallazgos se han trazado previamente con los atributos de calidad identificados en la codificación axial (ver Tabla 4.7), es importante destacar que también permiten comenzar a identificar otros atributos que son latentes en las respuestas: escalabilidad, interoperabilidad, estabilidad y usabilidad, entre otros. Estos atributos emergen no como requisitos específicos, sino como dimensiones transversales de calidad percibidas por los usuarios como fundamentales para el éxito de la solución.

Así, este análisis de frecuencias constituye un insumo clave no solo para la codificación axial y la matriz de trazabilidad temática, sino también como base empírica que orienta el trabajo posterior de identificación y priorización de atributos de calidad, abordado en el siguiente objetivo específico.

Contingencia Q6 × Q13 Para examinar la coherencia entre los problemas identificados por los usuarios (Q6) y las mejoras sugeridas (Q13), se realizó un cruce categórico que permitió construir una matriz de contingencia, cuyos residuos estandarizados se visualizan en la Fig. 4.6. Aunque el test global de independencia entre ambas variables no resultó significativo ($\chi^2 = 17{,}8$, gl $= 16$, $p = 0{,}33$), los residuos estandarizados permiten identificar asociaciones locales estadísticamente relevantes.

En particular, los residuos con valores absolutos $|r| \geq 1{,}96$ indican desviaciones significativas respecto a la hipótesis nula de independencia. Es decir, reflejan relaciones que aparecen con una frecuencia mayor o menor a la esperada por azar, y por tanto representan interacciones cualitativamente relevantes entre dificultades percibidas y propuestas de mejora. Este análisis refuerza el

principio de trazabilidad entre las necesidades expresadas por los usuarios y las soluciones propuestas en la arquitectura del sistema.

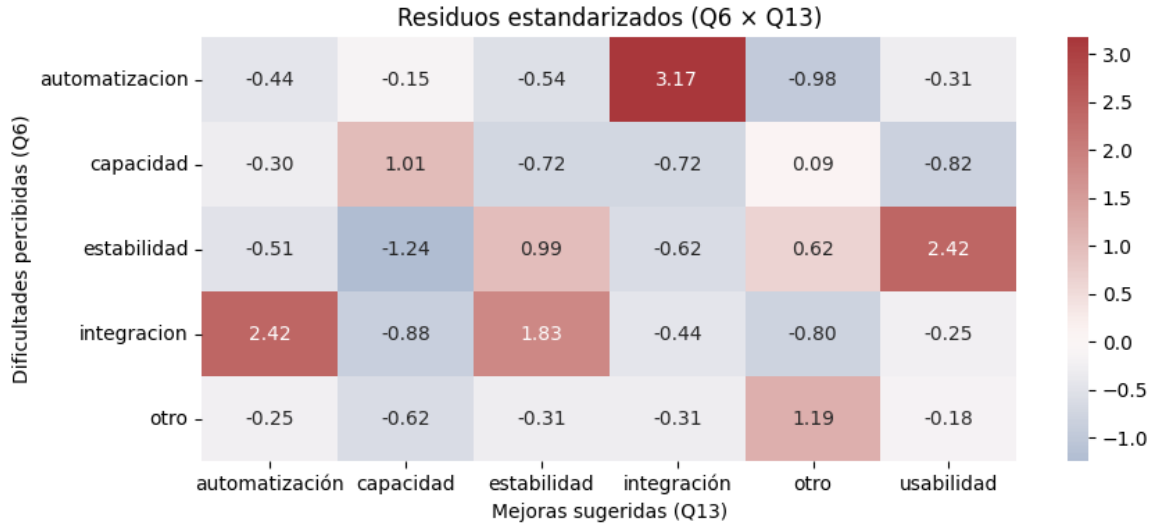


Figura 4.6: Matriz de residuos estandarizados en el cruce entre dificultades (Q6) y mejoras sugeridas (Q13).

Los tres cruces más destacados, por su significancia estadística e implicación arquitectónica, son los siguientes:

- **Automatización → Integración** ($r = +3{,}17$): evidencia una necesidad urgente de mejorar la interoperabilidad entre componentes del sistema, en particular mediante la integración de funcionalidades como el inicio único (SSO) y la vinculación automática con plataformas LMS. Este hallazgo cualitativo refuerza la pertinencia de los requisitos definidos en el bloque de *Integración con Plataformas Externas* y anticipa la importancia de atributos de calidad como la **compatibilidad** y la **interoperabilidad**.

- **Integración ↔ Automatización** ($r = +2{,}42$): señala una demanda recíproca de automatización de procesos críticos (como la creación masiva de enlaces o salas), con implicaciones directas en los requisitos priorizados en las secciones de *Administración del Sistema* y *Orquestación de Aforo*. Este patrón refuerza la necesidad de garantizar atributos como la **escalabilidad**, clave para sostener el funcionamiento en escenarios de alta demanda.

- **Estabilidad → Usabilidad** ($r = +2{,}42$): sugiere que las fallas en la estabilidad del sistema afectan significativamente la experiencia de uso, lo que da respaldo a los requisitos incluidos en el bloque de *Informes y Monitoreo* (como los que alertan sobre caídas y latencias) y valida la inclusión del atributo de calidad **usabilidad**, especialmente en contextos con conectividad

limitada. Esto se alinea con la necesidad de ofrecer versiones ligeras del sistema que garanticen accesibilidad amplia.

Estos hallazgos, además de afinar la priorización funcional (Cap. 5), orientan directamente hacia los atributos de calidad que serán desarrollados en la siguiente sección, como lo son la interoperabilidad, estabilidad, escalabilidad y usabilidad. Dichos atributos surgen no solo de la literatura o normas técnicas, sino de la experiencia empírica de los usuarios, fortaleciendo así el sustento contextual de la arquitectura propuesta.

Trazabilidad Tema → Atributo de calidad La Tabla 4.7 sintetiza la correspondencia entre las categorías axialmente consolidadas (producto del análisis cualitativo de las preguntas abiertas Q6 y Q13) y los atributos de calidad inferidos. Este ejercicio permite conectar de forma directa los hallazgos empíricos con las dimensiones no funcionales que deben guiar el diseño de la solución arquitectónica, facilitando así el cumplimiento del Objetivo Específico 2.

Tabla 4.7: Trazabilidad entre temas cualitativos y atributos de calidad.

Tema axial	**Evidencia típica (Q6/Q13)**	**Atributo de calidad inferido**
Capacidad	"Más de 1000 participantes conectados al mismo tiempo"	Escalabilidad horizontal, orquestación dinámica de aforo, elasticidad en el número de sesiones simultáneas.
Automatización	"Asignar salas automáticamente según el número de estudiantes"	Eficiencia operativa, reducción de errores humanos, capacidad de autoconfiguración.
Integración	"Conectar con Moodle / LMS institucional"	Interoperabilidad, compatibilidad con estándares, seguridad en la federación de identidades.
Estabilidad	"La plataforma se cae en los picos de uso"	Disponibilidad, tolerancia a fallos, continuidad del servicio en condiciones de alta demanda.
Usabilidad	"Es difícil saber dónde hacer clic para unirme a la sesión"	Experiencia de usuario, accesibilidad, claridad en los flujos de interacción.
IA / Funcionalidad futura	"Sería útil tener transcripción automática de las clases"	Capacidad de expansión funcional, adaptabilidad tecnológica, incorporación progresiva de inteligencia artificial.

La matriz fue construida manualmente a partir de la codificación axial consolidada en el cuaderno

`03_analisis_cualitativo.ipynb`, y será posteriormente formalizada en un archivo de trazabilidad versionado para su incorporación como insumo al diseño arquitectónico (etapa posterior).

En conjunto, estos hallazgos cualitativos no solo fortalecen la interpretación de las dificultades percibidas por los usuarios, sino que permiten abstraer atributos de calidad relevantes que deberán ser considerados en el modelo arquitectónico. La secuencia que sigue profundizará en estos atributos, detallando su definición, justificación empírica y relación con la solución esperada.

4.6. Resultados integrados del análisis mixto

Con el propósito de sintetizar los hallazgos obtenidos a partir del análisis cuantitativo y cualitativo, esta sección presenta una integración estructurada de los resultados empíricos, organizada en función de los temas emergentes, los patrones estadísticos y su correspondencia con atributos de calidad relevantes. La inclusión de esta síntesis permite establecer una conexión directa entre la evidencia recopilada y las decisiones de diseño arquitectónico que se abordan en la siguiente sección.

La convergencia metodológica derivada del diseño mixto empleado permite observar una serie de coincidencias y complementariedades entre ambas fuentes de análisis:

- La **capacidad del sistema** fue identificada como una limitación crítica tanto en los datos cualitativos (menciones directas a saturación de sesiones y restricciones de aforo) como en los datos cuantitativos (asociación transversal entre experiencia docente y percepción de insuficiencia). Este hallazgo respalda la inclusión de escalabilidad como atributo de calidad prioritario.

- La **automatización de procesos operativos** se destacó en el análisis cualitativo como una necesidad frente a la gestión masiva de sesiones. A nivel cuantitativo, la alta experiencia técnica de los usuarios encuestados refuerza la viabilidad de implementar funciones avanzadas sin incrementar la carga cognitiva, justificando así atributos como eficiencia operativa y autoconfiguración.

- La **integración con sistemas institucionales** fue mencionada en múltiples fragmentos cualitativos, principalmente en relación con la interoperabilidad con LMS. Dada la criticidad de esta funcionalidad y su potencial riesgo en términos de autenticación, se reconoce la necesidad de garantizar compatibilidad, federación de identidades y seguridad en las conexiones.

- La **estabilidad del sistema** fue referida en testimonios sobre caídas frecuentes en horarios pico, lo cual coincide con la necesidad de una arquitectura tolerante a fallos. El análisis cuantitativo, aunque no mostró diferencias significativas entre grupos, evidenció que la preocupación por estabilidad es compartida en toda la muestra.

- La **usabilidad** surgió como una problemática reiterada, especialmente en cuanto a la claridad de navegación. Esta preocupación, combinada con la alta alfabetización tecnológica de la muestra, indica que la interfaz debe ser intuitiva sin subestimar las capacidades del usuario avanzado.

- Finalmente, el tema transversal de **inteligencia artificial y funcionalidades futuras** fue planteado como un valor agregado deseable más que como una necesidad inmediata. Si bien no se codificó como requisito prioritario, sí establece una ruta potencial de expansión funcional en futuras versiones del sistema.

Con base en estas convergencias, se consolida la Tabla 4.8, la cual resume de manera estructurada los hallazgos combinados y los atributos de calidad derivados, organizados por eje temático.

Tabla 4.8: Resumen de hallazgos integrados y atributos de calidad asociados

Eje temático	**Hallazgos clave (mixtos)**	**Atributo(s) de calidad asociado(s)**
Capacidad	Saturación de sesiones, limitación de aforo, impacto transversal	Escalabilidad, orquestación de aforo, elasticidad
Automatización	Necesidad de asignación masiva, reducción de carga operativa	Eficiencia, autoconfiguración, bajo acoplamiento
Integración	Sincronización con LMS, login institucional, federación	Interoperabilidad, seguridad, compatibilidad
Estabilidad	Caídas del sistema, demanda en horarios críticos	Disponibilidad, tolerancia a fallos, continuidad del servicio
Usabilidad	Interfaz confusa, dificultad de acceso a enlaces	Accesibilidad, experiencia de usuario, claridad operativa
IA / Expansión	Deseo de transcripción automática y mejoras futuras	Evolutividad, adaptabilidad tecnológica, extensibilidad

Esta tabla cierra el proceso de análisis empírico y permite establecer una base argumentativa sólida para la definición, priorización y justificación de los atributos de calidad que orientarán la arquitectura propuesta en el capítulo siguiente.

Capítulo 5

Determinación de Atributos de Calidad

5.1. Objetivo y contexto

En ingeniería de software, los atributos de calidad (AQ) constituyen propiedades esenciales que determinan el éxito de un sistema informático más allá del mero cumplimiento funcional. Dichos atributos definen cómo debe comportarse el sistema bajo diversas condiciones operativas y cómo se alineará con las expectativas de sus usuarios y otros grupos de interés [60].

Esta sección aborda la identificación y determinación sistemática de los atributos de calidad prioritarios para el sistema de agendamiento multisala de videoconferencias, a partir de los hallazgos empíricos reportados en el capítulo previo. En particular, se enfatizan los atributos críticos (escalabilidad, disponibilidad, seguridad, usabilidad, interoperabilidad y extensibilidad) que resultaron más relevantes en el análisis cuantitativo y cualitativo desarrollado 4.

La relevancia de esta determinación radica en que, conforme a lo señalado por Bass et al.[60], los atributos de calidad impactan significativamente en las decisiones de diseño arquitectónico, condicionando la elección de patrones y estilos que posteriormente se implementarán. Por tanto, una definición rigurosa y fundamentada en el modelo ISO/IEC 25010:2023 garantiza que la arquitectura propuesta satisfaga las limitaciones y necesidades identificadas en plataformas como Zoom y Microsoft Teams, en términos de estabilidad, rendimiento, escalabilidad y seguridad [61, 62].

Este proceso se apoya en metodologías consolidadas: el enfoque Attribute-Driven Design (ADD) para la selección y priorización de atributos, y el Architecture Tradeoff Analysis Method (ATAM) para la evaluación de riesgos y compensaciones asociadas a cada AQ [60, 63]. Complementariamente, se adoptan técnicas visuales y de documentación que facilitan la comunicación efectiva de los resultados a todos los stakeholders [62].

En resumen, este apartado sienta el marco conceptual y metodológico para determinar con precisión los atributos de calidad críticos que guiarán las decisiones arquitectónicas del sistema de agendamiento, asegurando su alineación con los objetivos institucionales y operacionales definidos previamente.

5.2. Marco Teórico de Atributos de Calidad

La base conceptual de los atributos de calidad se sustenta en el modelo ISO/IEC 25010:2011, que define ocho características (funcionalidad, rendimiento, compatibilidad, usabilidad, fiabilidad, seguridad, mantenibilidad y portabilidad) y veintitrés subcaracterísticas, proporcionando un lenguaje común para la especificación y evaluación de requisitos de calidad [61].

Para integrar estos atributos en el diseño, Bass et al. (2012) describen el proceso Attribute-Driven Design (ADD), que emplea escenarios de atributo de calidad para derivar y priorizar requisitos arquitectónicos [60]. A su vez, el Architecture Tradeoff Analysis Method (ATAM) ofrece un enfoque sistemático para evaluar compensaciones entre atributos, identificando riesgos y puntos de sensibilidad mediante utility trees y escenarios de evaluación [63]. Richards & Ford (2020) complementan estos métodos con tácticas y patrones arquitectónicos aplicables a entornos distribuidos y de microservicios, ilustrando su uso en casos reales [62].

En fases tempranas de desarrollo, Gilson et al. (2019) demostraron cómo extraer atributos de calidad directamente de *user stories*, facilitando su incorporación a la definición de la arquitectura desde el primer iteration cycle [64]. Lytra et al. (2019) analizaron el uso de atributos de calidad en múltiples métodos de decisión arquitectónica, destacando buenas prácticas y puntos de mejora en su aplicación [65].

Para la modelación y evaluación cuantitativa de atributos, se han propuesto marcos como SQ-ME, que ofrece un meta-modelo para representar y medir atributos de calidad en distintas vistas arquitectónicas [66]; y metodologías basadas en análisis jerárquico difuso (FAHP), que permiten comparar y puntuar estilos arquitectónicos según criterios de calidad [67, 68]. Adicionalmente, técnicas de priorización como el *fuzzy MoSCoW* facilitan la ponderación rigurosa de atributos en contextos con incertidumbre [69].

Este conjunto de modelos, métodos y herramientas conforma el marco teórico sobre el cual se construirá la selección, priorización y evaluación de los seis atributos críticos de este estudio.

5.2.1. Modelado C4 y técnicas de diagramación moderna

Para documentar y comunicar de forma clara la arquitectura del Sistema de Agendamiento Multisala, adoptamos el *C4 model* de Simon Brown, que organiza la representación en cuatro vistas: Contexto (Nivel 1), Contenedores (Nivel 2), Componentes (Nivel 3) y Código (Nivel 4) facilitando la trazabilidad desde la visión global hasta los detalles de implementación [70].En este trabajo presentamos los tres primeros niveles:

1. El Diagrama de Contexto (Nivel 1), para delimitar el sistema, sus actores y sus dependencias externas.

2. El Diagrama de Contenedores (Nivel 2), donde se describen los servicios, aplicaciones, colas de mensajes y bases de datos que materializan nuestros atributos de calidad.

3. El Diagrama de Componentes (Nivel 3), que descompone internamente cada contenedor en sus unidades funcionales esenciales.

4. El Nivel 4 (detallado a nivel de clases o módulos de código) queda fuera del alcance de esta tesis, pues el objetivo es definir y justificar la arquitectura en términos de calidad y no profundizar en la implementación concreta.

Con este marco, garantizamos diagramas coherentes, actualizables y alineados con las decisiones de diseño basadas en los atributos de calidad identificados en el capítulo 4.

A continuación presentamos el primer artefacto C4: el Diagrama de Contexto (C4 – Nivel 1).

5.2.2. Diagrama de contexto (C4 – Nivel 1)

Para comprender el entorno en el que opera el sistema de agendamiento multisala, presentamos a continuación el diagrama de contexto (C4 – Nivel 1). Este diagrama muestra el *System Scope Boundary*, el actor principal y los sistemas externos con los que interactúa, facilitando la definición clara del alcance de nuestro diseño arquitectónico [60].

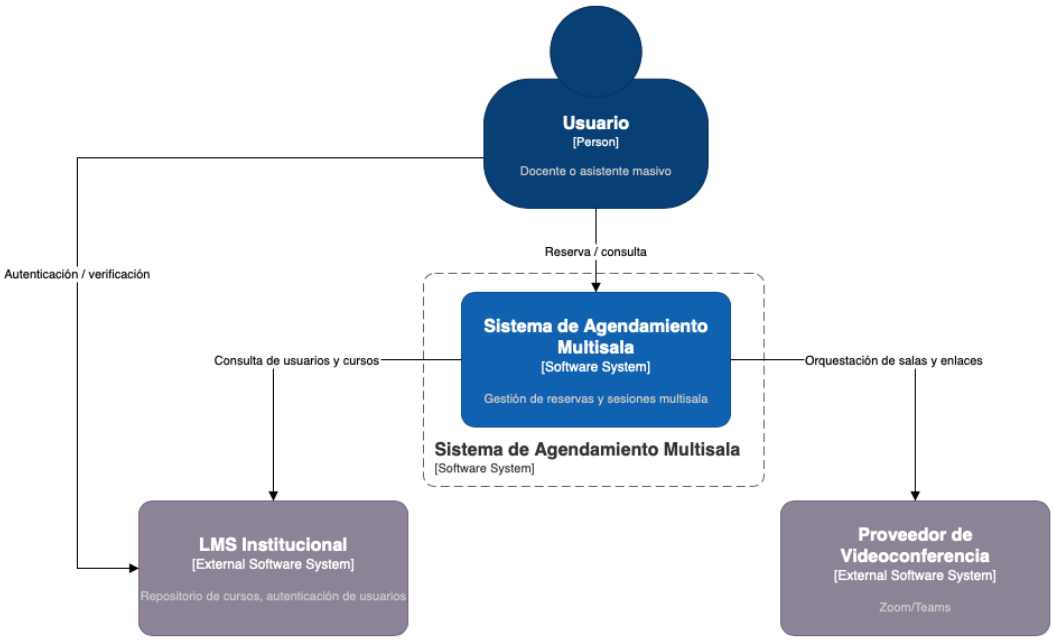


Figura 5.1: Diagrama C4 de contexto (Nivel 1): *Sistema de Agendamiento Multisala*, actor principal y sistemas externos

En la Figura 5.1 se aprecia que:

- El **Usuario** (docente o asistente masivo) es el único actor humano e interactúa con el sistema para reservar y consultar sesiones multisala.
- El **Sistema de Agendamiento Multisala** (Software System) centraliza la lógica de creación, orquestación y gestión de salas.
- El sistema se integra con un **LMS Institucional** (External Software System) para la autenticación y consulta de cursos/usuarios.
- El sistema se apoya en un **Proveedor de Videoconferencia** (External Software System, Zoom/Teams) para la realización efectiva de las sesiones.

Este diagrama de contexto establece de forma inequívoca los límites de nuestro sistema y los elementos externos relevantes, punto de partida esencial antes de detallar las subarquitecturas internas

(niveles C2–C4) y los atributos de calidad que deben guiar su diseño.

5.2.3. Diagrama de contenedores (C4 – Nivel 2)

Tras definir con claridad el alcance y los actores externos en el Diagrama de Contexto (Nivel 1, Figura 5.1), es necesario describir cómo se estructura internamente el *Sistema de Agendamiento Multisala*. El Diagrama de Contenedores (Nivel 2) muestra cada uno de los elementos desplegados (aplicaciones web, microservicios, colas y bases de datos) junto con sus tecnologías y las principales interacciones que facilitan la orquestación de reservas y la gestión de sesiones multisala.

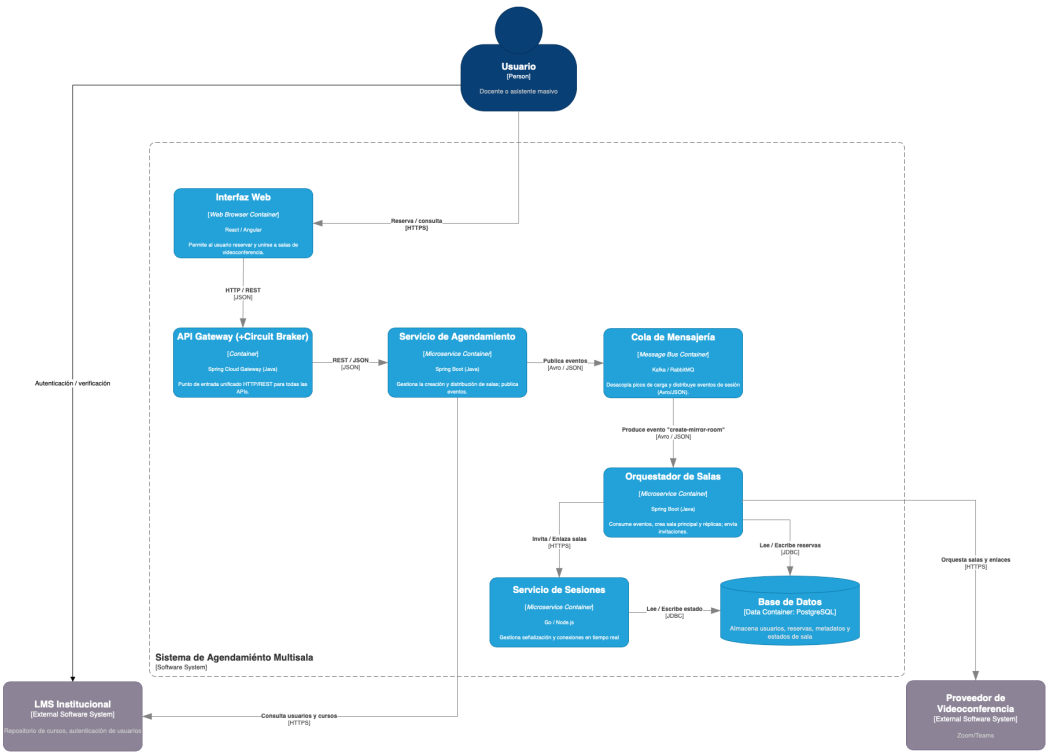


Figura 5.2: Diagrama C4 de contenedores (Nivel 2): componentes internos del Sistema de Agendamiento Multisala, mostrando tecnologías, boundary del sistema y sistemas externos.

En la Figura 5.2 se aprecia que:

- La **Interfaz Web** (*React/Angular*) que permite al usuario reservar y unirse a sesiones multisala directamente desde el navegador.

- El **API Gateway** (*Spring Cloud Gateway, Java*) como punto de entrada HTTP/REST, protegido por un circuito breaker que aisla instancias defectuosas y enruta las peticiones al servicio correspondiente.

- El **Servicio de Agendamiento** (*Spring Boot, Java*) encargado de validar usuarios y cursos, coordinar la creación de salas y publicar eventos de sesión en la cola de mensajería.

- La **Cola de Mensajería** (*Kafka/RabbitMQ*) que desacopla picos de carga y garantiza la entrega fiable de los eventos entre productores y consumidores.

- El **Orquestador de Salas** (*Spring Boot, Java*) que consume eventos, crea la sala principal y sus réplicas espejo, persiste los metadatos y envía las invitaciones diferenciadas.

- El **Servicio de Sesiones** (*Go/Node.js*) que gestiona la señalización WebRTC y las conexiones en tiempo real, desplegado en dos Availability Zones para alta disponibilidad.

- La **Base de Datos** (*PostgreSQL*) que almacena usuarios, reservas, metadatos y estados de sala, con replicación síncrona entre zonas (RPO = 0) para asegurar la durabilidad.

- El sistema se integra con un **LMS Institucional** (External Software System) para la autenticación federada y la consulta de cursos y usuarios.

- El sistema utiliza un **Proveedor de Videoconferencia** (External Software System, Zoom/-Teams) para la ejecución efectiva de las sesiones multisala.

Este diagrama de contenedores define claramente los servicios y las dependencias externas que conforman nuestro sistema. Constituye la base sobre la que se descompondrán en detalle los componentes internos (C4 – Nivel 3) y se evaluará cada elemento frente a los atributos de calidad definidos (escalabilidad, disponibilidad, seguridad, usabilidad, interoperabilidad y extensibilidad).

5.3. Priorización de Atributos

Este apartado retoma los seis atributos de calidad pre-seleccionados en el capítulo 3 (véanse Tabla 4.1, Tabla 4.6 y Tabla 4.7) y aplica dos enfoques complementarios para su priorización: una clasificación cualitativa MoSCoW y dos índices compuestos cuantitativos.

5.3.1. Clasificación MoSCoW de AQ

La matriz MoSCoW asigna a cada atributo de calidad uno de cuatro niveles de prioridad [69]:

- **Must have**: atributos imprescindibles para satisfacer las limitaciones críticas L1–L3 (capacidad, estabilidad y seguridad).
- **Should have**: atributos muy deseables, pero cuya ausencia no impide la operación básica.
- **Could have**: atributos opcionales que aportan valor adicional.
- **Won't have**: atributos fuera del alcance en esta fase.

Tabla 5.1: Clasificación MoSCoW de atributos de calidad (umbral difuso)

Must have (IPA, ISS $> 0{,}90$)	**Should have** ($0{,}70 \leq$ IPA $\leq 0{,}90$)	**Could have** ($0{,}60 \leq$ IPA $< 0{,}70$)	**Won't have** (IPA $< 0{,}60$)
Escalabilidad	Usabilidad	Extensibilidad	IA / Expansión futura
Disponibilidad	Interoperabilidad		
Seguridad			

Esta asignación se fundamenta en: (i) la correspondencia L1→Escalabilidad, L2→Disponibilidad y L3→Seguridad (Tabla 4.1); (ii) los valores de IPA e ISS calculados (Tabla 5.2), que sitúan a Escalabilidad, Disponibilidad y Seguridad por encima de 0.90; (iii) la frecuencia y relevancia de Usabilidad e Interoperabilidad señaladas en el análisis cualitativo (Tabla 4.7); (iv) el carácter exploratorio de IA/Expansión futura, sin métricas cuantitativas que justifiquen su prioridad inmediata. Con este criterio, la clasificación refleja tanto la evidencia empírica como los umbrales difusos de prioridad.

5.3.2. Selección de atributos críticos y trade-offs

De los seis atributos de calidad preseleccionados (Tabla 5.1), tres (*Escalabilidad*, *Disponibilidad* y *Seguridad*) clasifican como **Must have**, dado su impacto directo en las limitaciones críticas L1–L3.

Para fundamentar esta elección y guiar las decisiones arquitectónicas, construimos un pequeño *utility tree* ATAM que relaciona cada atributo con:

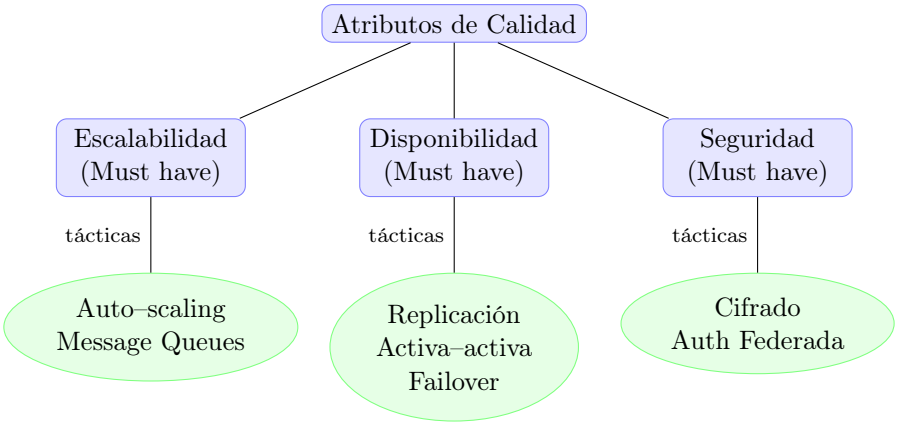


Figura 5.3: Utility tree ATAM para los atributos críticos.

- **Escalabilidad:** riesgo de saturación de sesiones en picos de demanda; tácticas de desempeño, colas y *auto-scaling*[1].
- **Disponibilidad:** riesgo por fallo de infraestructura (caída de AZ); *replicación activa–activa*[2], *failover automático*[3], *health checks*[4] y *circuit breaker*[5].
- **Seguridad:** riesgo de brechas y ataques maliciosos; cifrado, autenticación federada y auditoría continua.

Con este árbol de decisión documentamos:

1. Los riesgos primarios que afronta cada atributo.
2. Las tácticas de alto nivel escogidas para mitigarlos.
3. Los trade-offs inherentes (por ejemplo, priorizar disponibilidad puede penalizar ligeramente la interoperabilidad al introducir complejidad en la sincronización de réplicas).

A continuación, en los apartados 4.4 (Escalabilidad), 4.5 (Disponibilidad) y 4.6 (Seguridad) presentamos en detalle, para cada atributo:

[1] Mecanismo de *escalado* que eleva o reduce automáticamente el número de instancias de un servicio según métricas de carga (CPU, conexiones, latencia), sin intervención manual.

[2] Estrategia de alta resiliencia en la que dos (o más) instancias idénticas están operativas al mismo tiempo y comparten la carga, eliminando puntos únicos de fallo.

[3] Conmutación transparente al nodo secundario al detectar un fallo, cumpliendo un *Recovery Time Objective* (RTO) predefinido.

[4] Sondeos periódicos (HTTP/gRPC, etc.) que verifican la salud de cada instancia para poder retirarla del balanceo si falla.

[5] Patrón que abre un "interruptor" lógico cuando un servicio presenta demasiados errores consecutivos, evitando nuevas peticiones hasta su recuperación y protegiendo al sistema de fallos en cascada.

- Su definición y métricas relevantes (por ejemplo, MTBF/MTTR, SLA 99.99 %).
- Escenarios ATAM concretos y criterios de aceptación.
- Tácticas especializadas de detección, recuperación y prevención.
- Patrones arquitectónicos más comunes y sus compensaciones.

5.3.3. Métricas cuantitativas de priorización

Para complementar la priorización cualitativa (MoSCoW) y fundamentar numéricamente la selección de atributos, a continuación se introducen dos índices compuestos que cuantifican la criticidad de cada atributo de calidad.

Índice de Priorización de Atributos (*IPA*). La ecuación utilizada para calcular el ***IPA*** de cada atributo es:

$$\mathrm{IPA}_i \;=\; \frac{\sum_{j=1}^{n} w_j \, r_{ij}}{\sum_{j=1}^{n} w_j}$$

donde w_j representa el peso (difuso) asignado al criterio j y r_{ij} la valoración cuantitativa del atributo i en dicho criterio [68]. Este modelo Fuzzy–AHP pondera tanto la severidad operativa como la importancia percibida por los stakeholders.

Índice de Severidad de Software (*ISS*). El ***ISS*** se basa en la integral difusa propuesta por Elahi & Babamir [67], combinando la frecuencia de fallos y la criticidad percibida para normalizar la severidad de cada atributo en el rango $[0, 1]$.

Tabla 5.2: Valores de IPA e ISS y clasificación de criticidad[6]

Atributo	IPA	ISS	Prioridad
Escalabilidad	0.90	0.88	Alta
Disponibilidad	0.95	0.92	Muy alta
Seguridad	0.93	0.94	Muy alta
Usabilidad	0.75	0.60	Media
Interoperabilidad	0.70	0.55	Media–baja
Extensibilidad	0.65	0.50	Baja

Estos resultados confirman y matizan la clasificación MoSCoW: mientras que **Escalabilidad**, **Disponibilidad** y **Seguridad** alcanzan los valores más altos de IPA e ISS, **Usabilidad** e **Interoperabilidad** quedan en niveles intermedios y **Extensibilidad** se considera de menor urgencia en la fase inicial. A partir de esta cuantificación, se refuerza la elección de las tácticas y patrones arquitectónicos presentados en los apartados siguientes.

[6] Valores redondeados a dos decimales. El atributo "IA / Expansión futura" no se incluye porque no se disponía de datos cuantitativos.

5.4. Escalabilidad

La *escalabilidad* es la capacidad de un sistema para mantener niveles de rendimiento aceptables al incrementarse la carga de trabajo. Para nuestro escenario ATAM (Sección 5.3.2), buscamos atender +27.000 usuarios simultáneos con un tiempo de respuesta (t_{resp}) inferior a 2 s y una creación automática de salas principal y espejo en menos de 5 s.

5.4.1. Definición y métricas relevantes

Entre las métricas clave para medir la escalabilidad se encuentran:

- **Throughput:** número de transacciones (creación de salas) por segundo.
- **Latency:** tiempo de respuesta medio por petición (t_{resp}).
- **Elasticidad:** capacidad de crecer y decrecer instancias sin intervención manual.

Estos indicadores se reflejan en nuestro *IPA* (0.90) e *ISS* (0.88), según la Tabla 5.2.

5.4.2. Escenario ATAM y criterios de aceptación

Definimos el siguiente escenario [63]:

> *"27.000 usuarios simultáneos, $t_{\text{resp}} < 2\,s$, creación automática de la sala principal y dos salas espejo, con envío de enlaces diferenciados en $< 5\,s$."*

Los criterios de aceptación asociados son:

- Mantener un throughput de al menos 4.000 eventos/s.
- Garantizar $t_{\text{resp}} < 2$ s en el 95 % de las peticiones.
- Autoescalado que incorpore o retire instancias en menos de 60 s.
- Límites de cola que eviten pérdida de eventos o caídas del sistema.

5.4.3. Tácticas especializadas

Para mitigar el riesgo de saturación de sesiones [60, cap. 9], empleamos:

- **Manage event rate:** control de ráfagas limitando peticiones simultáneas.
- **Prioritize events:** atender primero los eventos de creación de sala principal.
- **Auto–scaling groups:** añadir o quitar instancias del Servicio de Sesiones según métricas de CPU y latencia [62].

- **Maintain multiple copies:** replicar el Servicio de Sesiones en múltiples nodos para distribuir la carga.
- **Message queues:** desacoplar el Servicio de Agendamiento y el Servicio de Sesiones con Kafka/RabbitMQ.
- **Bound queue sizes:** fijar umbrales para detectar y gestionar ráfagas extremas.

5.4.4. Patrones arquitectónicos y compensaciones

Los patrones adoptados y sus trade–offs principales son:

- **Microservicios + colas:** alta desacoplo y elasticidad, a costa de mayor complejidad de orquestación.
- **Auto–scaling:** respuesta automática a la demanda, pero requiere calibración de políticas y aumenta la latencia de arranque.
- **Replicación de servicios:** reduce cuellos de botella, pero complica la consistencia en tiempo real.
- **Colas con límite:** protegen contra sobrecarga, pero pueden retrasar mensajes si no se gestionan adecuadamente.

5.4.5. Escenario de Calidad para Escalabilidad

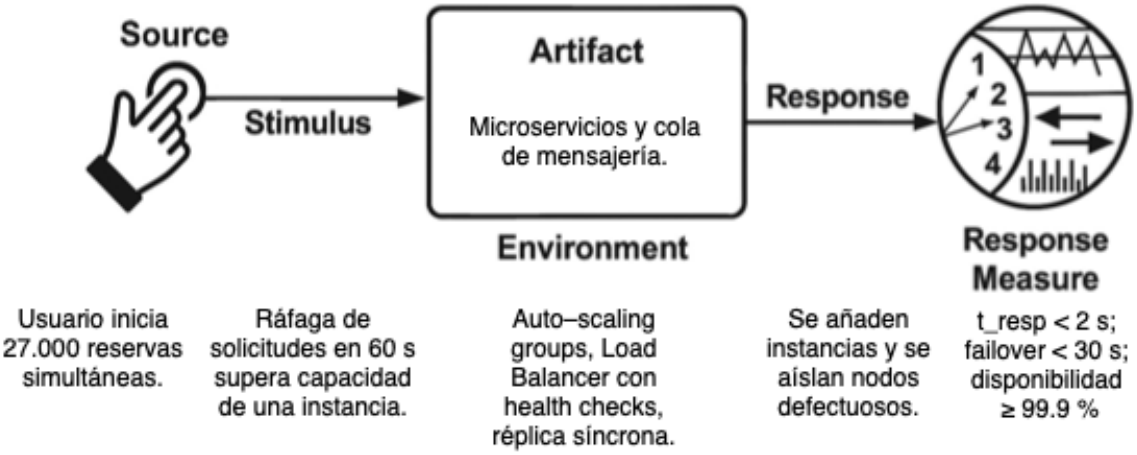


Figura 5.4: Escenario de calidad (ATAM) para Escalabilidad: **Source**, **Stimulus**, **Artifact**, **Environment**, **Response** y **Response Measure**.

En la Figura 5.4 se detallan los seis elementos del escenario:

Source: Usuario inicia 27 000 reservas simultáneas.

Stimulus: Ráfaga de solicitudes en 60 s supera la capacidad de una instancia.

Artifact: Microservicios y cola de mensajería.

Environment: Auto–scaling groups, Load Balancer con health checks y réplica síncrona.

Response: Se añaden instancias y se aíslan nodos defectuosos.

Response Measure: $t_{\text{resp}} < 2$ s; Failover < 30 s; disponibilidad $\geq 99{,}9\,\%$.

Con este escenario de calidad definimos y validamos las respuestas del sistema ante picos extremos de carga. Gracias al autoescalado horizontal y al aislamiento automático de fallos.

5.5. Disponibilidad

La *disponibilidad* es la capacidad de un sistema para permanecer operativo y accesible ante fallos de componentes o de la infraestructura subyacente. Según nuestro escenario ATAM (Sección 5.3.2), buscamos un *Recovery Time Objective* (RTO) inferior a 30 s y un *Recovery Point Objective* (RPO) igual a 0, garantizando continuidad sin pérdida de datos.

5.5.1. Definición y métricas relevantes

Las métricas clave para evaluar la disponibilidad son:

- **MTBF (Mean Time Between Failures):** tiempo medio entre fallos.
- **MTTR (Mean Time To Repair):** tiempo medio para recuperar el servicio.
- **SLA (Service Level Agreement):** p.ej. 99.99 % de uptime.

Estos valores se reflejan en nuestro Índice de Priorización de Atributos (IPA = 0.95) e Índice de Severidad de Software (ISS = 0.92) (Tabla 5.2).

5.5.2. Escenario ATAM y criterios de aceptación

Definimos el siguiente escenario [63]:

> *"Failover en menos de 30 s tras la caída de una AZ, sin pérdida de datos (RPO = 0)."*

Los criterios de aceptación son:

- Conmutación automática antes de 30 s tras fallo de cualquier instancia o AZ.
- Replicación síncrona de la base de datos que garantice RPO = 0.
- Aislamiento inmediato de instancias defectuosas tras un solo health check fallido.

5.5.3. Tácticas especializadas

Para alcanzar los objetivos anteriores, empleamos estas tácticas [60, cap. 10]:

- **Replicación activa–activa:** despliegue simultáneo de instancias en dos AZ.
- **Failover automático:** conmutación transparente al nodo secundario al detectar fallo.
- **Health checks continuos:** supervisión periódica de cada instancia.
- **Circuit breakers:** aislamiento rápido de instancias defectuosas.
- **Persistencia redundante:** réplica síncrona de la base de datos (RPO = 0).

5.5.4. Patrones arquitectónicos y compensaciones

Los patrones y sus principales trade–offs son:

- **Cluster activo–activo:** máxima resiliencia, a costa de complejidad de sincronización.
- **Load Balancer con health checks:** conmutación rápida, añade latencia mínima.
- **Circuit breaker en la capa de entrada:** protege de fallos puntuales; requiere ajuste fino de umbrales.
- **Base de datos con réplica síncrona:** RPO = 0, introduce latencia en confirmación de escritura.

5.5.5. Escenario de calidad para Disponibilidad

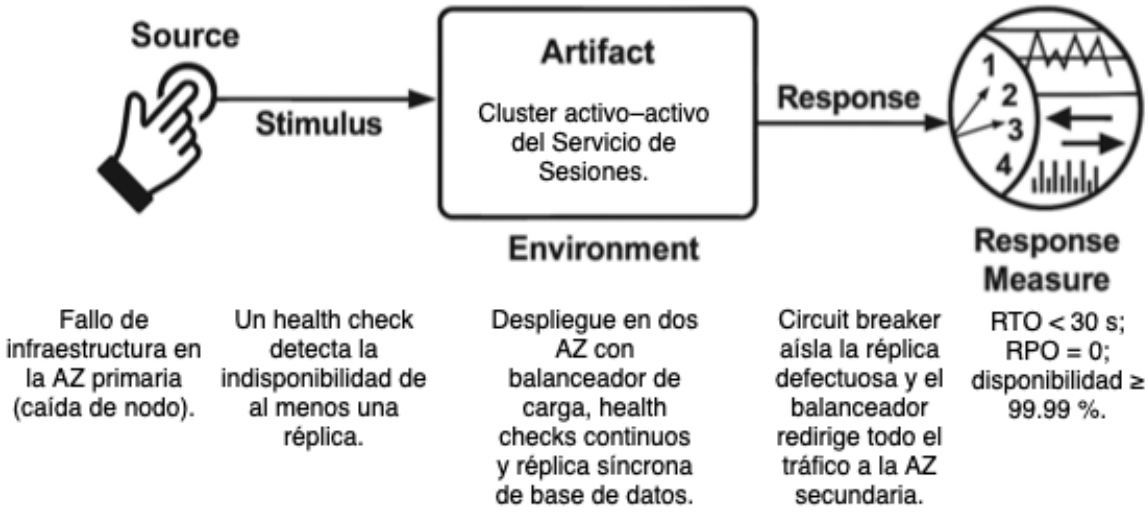


Figura 5.5: Escenario de calidad (ATAM) para Disponibilidad: **Source**, **Stimulus**, **Artifact**, **Environment**, **Response** y **Response Measure**.

En la Figura 5.5 se describen los elementos del escenario:

Source: Fallo de infraestructura en la AZ primaria (caída de nodo).

Stimulus: Un health check detecta la indisponibilidad de la réplica principal.

Artifact: Cluster activo–activo del Servicio de Sesiones.

Environment: Despliegue en dos AZ con Load Balancer, health checks continuos y réplica síncrona de base de datos.

Response: Circuit breaker aísla la réplica defectuosa y el Load Balancer redirige todo el tráfico a la AZ secundaria.

Response Measure: RTO < 30 s; RPO $= 0$; disponibilidad $\geq 99{,}9\,\%$.

Con este escenario de calidad documentamos:

1. El riesgo principal (caída de AZ).
2. Las tácticas de detección y recuperación (health checks, circuit breaker, failover).
3. Los criterios cuantitativos de éxito (RTO, RPO, SLA).

A continuación, en la Sección 5.6 profundizaremos en el atributo de Seguridad usando el mismo esquema de definición, escenario, tácticas y patrones.

5.6. Seguridad

La *seguridad* es la capacidad de un sistema para protegerse contra accesos no autorizados, brechas de datos y ataques maliciosos, garantizando confidencialidad, integridad y disponibilidad de la información. En nuestro escenario ATAM (Sección 5.3.2) establecemos un objetivo de mitigación de ataques comunes (XSS, DoS, brechas de datos) en al menos el 99.99 % de los intentos.

5.6.1. Definición y métricas relevantes

Entre las métricas clave para evaluar la seguridad se incluyen:

- **Número de vulnerabilidades críticas**: halladas en pruebas de pentesting.
- **Tasa de mitigación de ataques**: porcentaje de ataques bloqueados exitosamente.
- **Tiempo medio de detección (MTTD)** y **tiempo medio de respuesta (MTTR)** ante incidentes.
- **Cumplimiento de estándares**: p.ej. OWASP Top 10, ISO 27001.

En nuestra priorización obtenemos IPA = 0.93 e ISS = 0.94 (Tabla 5.2), lo que justifica su clasificación como "Must have".

5.6.2. Escenario ATAM y criterios de aceptación

Definimos el siguiente escenario [63]:

> *"Bloqueo de 99.99 % de intentos de XSS/DoS y brechas; detección y contención en menos de 5 s."*

Los criterios de aceptación asociados son:

- Todas las solicitudes maliciosas conocidas (OWASP Top 10) deben ser bloqueadas en la puerta de entrada.
- Tiempo medio de detección y contención de incidentes (MTTD/MTTR) inferior a 5 s.
- Registro de auditoría completo de todos los accesos fallidos y cambios de privilegios.

5.6.3. Tácticas especializadas

Para satisfacer estos criterios, empleamos las siguientes tácticas [60, cap. 11]:

- **Authentication and authorization:** federación de identidades con OAuth/OpenID Connect y control de accesos basado en roles (RBAC).

- **Input validation and sanitization:** filtrado y escape de todas las entradas para prevenir XSS e inyección de código.
- **Encryption in transit and at rest:** TLS para todas las comunicaciones y cifrado AES-256 de datos sensibles en la base de datos.
- **Audit logging:** registro inmutable de eventos de seguridad en un sistema de log centralizado.
- **Intrusion detection and rate limiting:** detección de patrones anómalos y limitación de tasa para mitigar DoS.

5.6.4. Patrones arquitectónicos y compensaciones

Los patrones adoptados y sus principales trade–offs son:

- **API Gateway con WAF:** bloquea ataques en la puerta de entrada; introduce un punto único de inspección y posible latencia adicional.
- **Autenticación federada (OAuth/OIDC):** mejora la gestión de credenciales; requiere integración con IdP externo.
- **Cifrado de datos en reposo:** asegura confidencialidad; añade sobrecarga de CPU en operaciones de E/S.
- **Logging centralizado e inmutable:** facilita auditoría forense; demanda almacenamiento y puede impactar el rendimiento si no se dimensiona bien.

5.6.5. Escenario de calidad para Seguridad

En la Figura 5.6 se describen los seis elementos del escenario:

Source: Agente atacante externo o interno malintencionado.

Stimulus: Intento de XSS, DoS o acceso no autorizado a recursos críticos.

Artifact: API Gateway con WAF y microservicios con validación estricta.

Environment: OAuth/OIDC para autenticación, TLS en todas las comunicaciones y cifrado AES-256 en reposo.

Response: Peticiones maliciosas bloqueadas; sesión atacada terminada; eventos registrados.

Response Measure: $\geq 99{,}9\,\%$ de ataques bloqueados; MTTD/MTTR < 5 s; auditoría completa de incidentes.

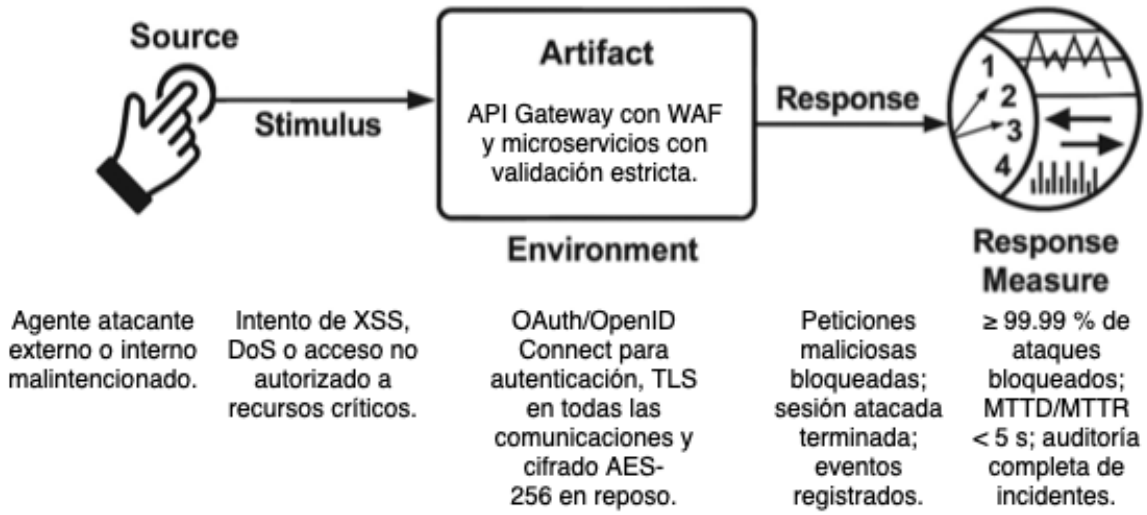


Figura 5.6: Escenario de calidad (ATAM) para Seguridad: **Source**, **Stimulus**, **Artifact**, **Environment**, **Response** y **Response Measure**.

Con este escenario garantizamos la detección, contención y registro de incidentes de seguridad dentro de los umbrales definidos, cerrando el ciclo de diseño para la limitación crítica de seguridad (L3).

5.6.6. Evaluación multi-atributo

Los tres escenarios ATAM definidos en las Seccs. 5.4, 5.5 y 5.6 se analizarán de forma *integrada*: un mismo árbol de utilidad y su matriz escenario–táctica permiten valorar las interacciones entre Escalabilidad, Disponibilidad y Seguridad sin construir prototipos adicionales.

La literatura respalda este enfoque. Mårtensson [71] demostró que es posible evaluar simultáneamente desempeño, mantenibilidad, portabilidad y testabilidad mediante un único protocolo experimental, lo cual valida la viabilidad de la evaluación multi-atributo en contextos industriales sin recurrir a un PoC ni un MVP.

Siguiendo esa evidencia, los resultados de los tres escenarios se consolidarán en la Tabla 5.3 antes de derivar las recomendaciones finales.

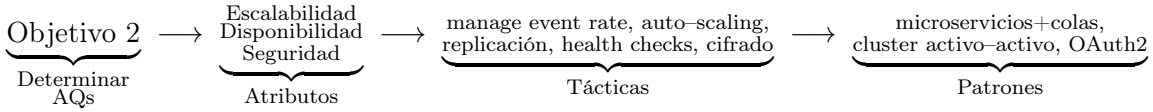
5.7. Resumen de Atributos de Calidad y Trazabilidad

Para cerrar el capítulo, la Tabla 5.3 consolida los tres atributos de calidad clasificados como *Must have* (Escalabilidad, Disponibilidad y Seguridad) junto con las tácticas, patrones y trade-offs más relevantes identificados durante el análisis.

Tabla 5.3: Resumen de atributos de calidad, tácticas, patrones y trade-offs

Atributo	**Tácticas clave**	**Patrones arquitectónicos**	**Trade-offs principales**
Escalabilidad	Manage event rate; Prioritize events; Auto–scaling groups; Message queues; Maintain multiple copies	Microservicios + colas; Auto–scaling groups; Réplicas de servicios	Mayor complejidad de orquestación; latencia de arranque en escalado
Disponibilidad	Replicación activa–activa; Failover automático; Health checks; Circuit breaker; Persistencia redundante	Cluster activo–activo; Load Balancer con health checks; Circuit breaker; Base de datos síncrona	Complejidad de sincronización; ligera latencia en confirmación; configuración de umbrales
Seguridad	Cifrado de datos en tránsito y reposo; Autenticación federada; Auditoría continua; WAF/API Gateway hardening	OAuth2/OpenID Connect; TLS mutuo; API Gateway + WAF; Logs centralizados	Overhead de cifrado; gestión de certificados; latencia mínima de autenticación

El siguiente mapa de trazabilidad sintetiza la cadena que parte del **Objetivo 2** y conecta los hallazgos del Capítulo 4 con las decisiones arquitectónicas sustentadas en las prácticas recomendadas por Bass et al. (*Software Architecture in Practice*, cap. 20 ADD y cap. 21 ATAM)[60, cap. 20–21]:



Este cierre garantiza la trazabilidad completa:

- De los **hallazgos empíricos** del Capítulo 4 (limitaciones L1–L3).
- A la **priorización MoSCoW** y los **índices IPA/ISS** (Sección 5.3).
- A los **escenarios ATAM** y **utility tree** (Sección 5.3.2).
- Hasta la **selección de tácticas y patrones** que definirán la arquitectura del Capítulo 5, conforme a la guía ADD/ATAM de Bass et al.[60, cap. 20–21].

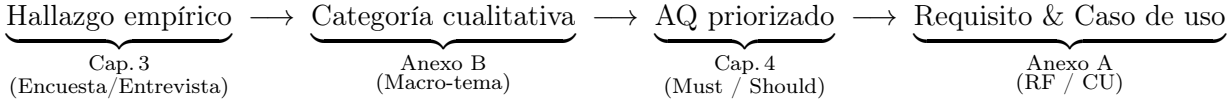
Con ello finaliza la determinación de atributos de calidad y se establece la base para la propuesta arquitectónica que se desarrolla en el capítulo siguiente.

5.8. Vínculo con los Anexos A y B

El Capítulo 5 identificó, priorizó y documentó los atributos de calidad *Escalabilidad*, *Disponibilidad* y *Seguridad*, así como las tácticas y patrones que los satisfacen (5.3,5.4,5.5,5.6).

- **Anexo A** (Desarrollo del proyecto, 9.1) integra esos atributos en la derivación de requisitos, diagramas de casos de uso y modelo de arquitectura. Para cada bloque funcional se indica, en la primera columna de las tablas de requisitos, la etiqueta *AQ-X* que enlaza con los atributos priorizados en 5.3.2.
- **Anexo B** (Tablas de categorización cualitativa) documenta las fuentes empíricas que originaron dichos atributos. La recategorización final de dificultades (10.3) incorpora una columna *AQ derivado* que traza cada macro-tema con los atributos de calidad seleccionados.

  De esta forma se preserva la trazabilidad:



Por lo anterior, el capítulo 5 se apoya en los **hallazgos empíricos** detallados en 10.3 (Anexo B) para derivar los **atributos de calidad** priorizados en 5.3.2. Dichos atributos se traducen luego en **requisitos funcionales** (RF-GV, RF-AS, . . .) y **casos de uso** del Anexo A (9.1, 9.4), manteniendo la trazabilidad Hallazgo → Categoría → AQ → Requisito.

Capítulo 6

Análisis De Plataformas En La Nube

6.1. Introducción

La adopción de *computación en la nube* constituye una decisión arquitectónica estratégica que impacta directamente los tres atributos de calidad priorizados en el Capítulo 5: **Escalabilidad**, **Disponibilidad** y **Seguridad**. De acuerdo con la definición de Microsoft, la nube es «una red global de centros de datos que alojan servicios de informática bajo demanda, accesibles a través de Internet y facturados bajo un modelo *pay–as–you–go*» [72]. Amazon Web Services (AWS) amplía ese concepto al subrayar que la nube permite «reemplazar gastos de capital por gastos operativos elásticos, pagando sólo por los recursos efectivamente utilizados» [73].

Bajo esta premisa, la elección de la plataforma nube más adecuada para el *Sistema de Agendamiento Multi-Sala de Videoconferencias* se convierte en un requisito no funcional de primer orden: la solución debe escalar hasta 27 000 usuarios simultáneos, garantizar un *uptime* no menor al 99,9 % y, al mismo tiempo, optimizar el coste total de propiedad[1]. Para cubrir estos objetivos, el presente capítulo compara de forma sistemática los servicios administrados de **Amazon Web Services** y **Microsoft Azure**, poniendo énfasis en:

1. *Cobertura global y resiliencia*: **37** regiones con **117** zonas de disponibilidad en AWS [74] frente a más de **60** regiones de Azure [75].
2. *Mecanismos de elasticidad*: *AWS Auto Scaling*, que permite lanzar o terminar instancias EC2 de forma automática [76], y *Azure Auto Scale*, que ajusta el número de máquinas virtuales según la demanda [77].
3. *Modelo de precios*: comparativa detallada de los esquemas *on-demand*, instancias reservadas, *Savings Plans* de AWS y *Azure Hybrid Benefit*/instancias reservadas de Azure.

Siguiendo la metodología recomendada por Richards & Ford para la toma de decisiones arquitectónicas [62], se evaluarán los servicios gestionados que nuestra arquitectura requiere (balanceadores, colas de mensajes, bases de datos relacionales y almacenamiento de objetos) bajo tres criterios cuantificables:

[1] Según Bass et al., el coste es un atributo de calidad que «limita o habilita (en la práctica) el nivel de los demás atributos» [60, cap. 4].

- **Costo mensual proyectado** para la carga de trabajo descrita en el Escenario ATAM de escalabilidad (§ 5.4.5).

- **Impacto en la latencia y el throughput** de la aplicación.

- **Nivel de esfuerzo operativo** (operación ↔ desarrollo), medido por la cantidad de tareas de aprovisionamiento y monitorización que el proveedor asume como servicio administrado.

La comparativa culmina con una recomendación basada en *costo-efectividad*: si bien ambos proveedores cubren los requisitos funcionales y no funcionales, la *economía de escala* y la variedad de planes de ahorro de AWS tienden a ofrecer un coste más competitivo para cargas de trabajo genéricas, incluyendo aquellas con picos abruptos de demanda [73]. No obstante, los casos en que una carga opera principalmente sobre licencias Microsoft (Windows Server, SQL Server) podrían beneficiarse del *Azure Hybrid Benefit*, lo que subraya la necesidad de un análisis detallado para cada caso de uso.

Los apartados siguientes profundizan en la descripción de ambos proveedores (Secciones 6.4 y 6.5), presentan la comparativa de servicios equivalentes y concluyen con un estudio de costos que respalda la decisión final de plataforma.

6.2. Metodología de comparación

El objetivo de esta sección es detallar cómo se obtendrán y analizarán los datos que alimentan la comparativa de Amazon Web Services (AWS) y Microsoft Azure. Con ello aseguramos la trazabilidad empírica entre los hallazgos del Capítulo 4, la priorización de atributos del Capítulo 5 y la tabla comparativa de la Sección 6.6.1.

6.2.0.1. Criterios y métricas de evaluación

Latencia (ms). Datos obtenidos de los benchmarks de red y de los white papers oficiales de cada proveedor [78, 79]. Se mide el tiempo medio de ida y vuelta (RTT) desde la región de referencia `us-east-1` para AWS y `East US` para Azure, hasta un nodo cliente ubicado en Bogotá (Colombia). **AQ evidenciado**: Escalabilidad / Desempeño.

Uptime (SLA). Probabilidad contractual de disponibilidad mensual publicada en los acuerdos de nivel de servicio [80, 81]. **AQ evidenciado**: Disponibilidad.

Costo mensual (USD). Estimaciones realizadas con las calculadoras oficiales [82, 83] para un clúster de referencia de 6 × vCPU-4 GiB, uso continuo (730 h/mes), instancia Linux, almacenamiento EBS/SSD 100 GB y transferencia saliente 3 TB/mes. No se incluyen tiers gratuitos ni créditos académicos. **AQ evidenciado**: Rentabilidad (vinculada a Escalabilidad y Disponibilidad).

Soporte TLS 1.3 e ISO 27001. Se verifica la disponibilidad de cifrado TLS 1.3 por defecto y la certificación ISO 27001 del servicio gestionado. [84]. **AQ evidenciado**: Seguridad.

Esfuerzo operativo. Número de tareas de aprovisionamiento, parcheo y escalado que el proveedor asume como servicio gestionado (valor ordinal: Bajo / Medio / Alto). **AQ evidenciado**: Mantenibilidad / Eficiencia operativa.

Tabla 6.1: Relación entre las métricas de comparación y los atributos de calidad prioritarios

Métrica	**Atributo ISO/IEC 25010**	**Justificación**
Latencia (RTT)	Rendimiento / Escalabilidad	A menor RTT, mejor desempeño percibido y menor tiempo de respuesta bajo carga crítica.
Uptime (SLA)	Disponibilidad	Probabilidad contractual de continuidad operativa; correlaciona con MTBF/MTTR.
Costo mensual	Rentabilidad (coste → Escalabilidad & Disponibilidad)	El presupuesto limita la cantidad de réplicas y mecanismos de failover que pueden desplegarse.
TLS 1.3 + ISO 27001	Seguridad	Garantiza confidencialidad en tránsito y gestión certificada de la información.
Esfuerzo operativo	Mantenibilidad / Eficiencia	Menos tareas manuales reducen riesgo humano y facilitan el escalado.

Con esta metodología aseguramos que cada dato comparativo se vincule directa y trazablemente con los atributos de calidad prioritarios definidos en los Capítulos 4 y 5. De esta forma, la tabla comparativa de la Sección 6.7.1 deja de ser una colección de cifras aisladas y se integra en el argumento central de la tesis.

6.3. Computación en la Nube: Conceptos Básicos

6.3.1. Definición y características de la nube

La *computación en la nube* puede entenderse, de manera general, como la *entrega bajo demanda de recursos informáticos a través de Internet con un modelo de pago por uso* [85]. Esta definición proporcionada por Amazon Web Services (AWS) destaca tres elementos clave: (i) la provisión de *recursos de TI* (cómputo, almacenamiento, redes, bases de datos, servicios de IA, entre otros); (ii) la disponibilidad *bajo demanda* (sin aprovisionamiento manual de infraestructura física); y (iii) un *esquema de costes elástico* basado en el consumo real.

Microsoft Azure complementa esta perspectiva al caracterizar la nube como una plataforma que *ofrece servidores, almacenamiento, bases de datos, analítica e inteligencia "para lograr una innovación más rápida, recursos flexibles y economías de escala"* [86]. Ambos proveedores coinciden en dos ideas rectoras:

- **Elasticidad y escalado rápido**: Los recursos pueden incrementarse o reducirse de forma automática, respondiendo a la demanda sin sobreaprovisionar infraestructura propia. AWS lo denomina *Elasticity* [85], mientras Azure habla de *recursos flexibles* AzureCloudComputing.

- **Modelo *pay-as-you-go***: Se sustituyen inversiones de capital (CAPEX) en centros de datos por gastos variables (OPEX), pagando solo por el tiempo y la capacidad realmente utilizados [85, 86].

A partir de estas definiciones y de la documentación técnica de ambas plataformas, pueden derivarse las siguientes características canónicas de la nube:

1. **Autoservicio bajo demanda**. Los consumidores aprovisionan recursos sin interacción humana con el proveedor (AWS Console, `CLI`, Azure Portal).

2. **Acceso ubicuo mediante red**. Los servicios están disponibles a través de mecanismos estándar (HTTP/HTTPS, API REST), garantizando accesibilidad global.

3. **Agrupamiento de recursos (*resource pooling*)**. La infraestructura del proveedor se comparte de forma multi-tenant, asignando y reasignando recursos de manera transparente.

4. **Elasticidad rápida**. Capacidad para escalar horizontal o verticalmente en minutos AWS destaca la "provisión en minutos" y Azure la "innovación acelerada" [85, 86].

5. **Servicio medido**. Uso monitorizado, controlado y facturado mediante métricas como horas-vCPU, GB-mes o peticiones por segundo.

6. **Amplia cobertura geográfica**. Posibilidad de *"desplegar globalmente en minutos"*, con regiones y zonas de disponibilidad distribuidas (AWS) o *Azure Regions*, lo que reduce latencia y facilita el cumplimiento de requisitos de soberanía de datos.

Estas propiedades constituyen el fundamento técnico que, en el Capítulo 6.1, permite comparar Amazon Web Services y Microsoft Azure bajo criterios de *costo total de propiedad (TCO)*, *elasticidad*, *rendimiento* y *disponibilidad regional*, buscando la opción más idónea para soportar la arquitectura de microservicios definida en el Capítulo 7.

6.3.2. Modelos de implementación (pública, privada, híbrida)

La tipología clásica de Nube definida por nist-800-145 distingue *pública*, *privada* e *híbrida* en función del modelo de propiedad y el grado de acceso. Tanto Amazon Web Services (AWS) como Microsoft Azure adoptan esta clasificación en su documentación oficial aws-cloud-models,azure-cloud-models. En el contexto de esta investigación, los tres modelos se valoran de acuerdo con los atributos prioritarios identificados en el Capítulo 5, en particular Escalabilidad (AQ-ESCAL), Disponibilidad (AQ-DISP) y Seguridad (AQ-SEC), así:

Nube pública Recurso multi-tenant ofrecido por el proveedor (`us-east-1` en AWS o `East US` en Azure). Destaca por *elasticidad bajo demanda* y **AQ-ESCAL**. Sin embargo, la dependencia total del enlace WAN introduce un punto de fallo que condiciona la **AQ-DISP**. La seguridad se rige por el modelo de responsabilidad compartida oficial.

Nube privada Infraestructura dedicada en las instalaciones del cliente o en un centro de datos exclusivo. Facilita el control sobre la capa física y lógicas de aislamiento, reforzando la **AQ-SEC**. No obstante, la capacidad queda limitada al tamaño del clúster local, lo que restringe la **AQ-ESCAL** y eleva el costo OPEX/CAPEX.

Nube híbrida Combina ambos mundos mediante servicios gestionados tales como *AWS Outposts* o *Azure Stack HCI* aws-cloud-models,azure-cloud-models. Permite mover cargas entre on-premises y la nube pública, atenuando los riesgos de disponibilidad (fail-over geográfico) y manteniendo datos sensibles in-house. En consecuencia, optimiza la combinación de **AQ-ESCAL**, **AQ-DISP** y **AQ-SEC**, a costa de mayor complejidad operativa y de red.

Rastreo con los capítulos previos. La elección entre estos modelos influye directamente en los escenarios ATAM de Escalabilidad y Disponibilidad definidos en las Secciones 5.4.5 & 5.5.5. Por ejemplo, un despliegue híbrido posibilita escalado horizontal hacia la Nube pública durante los picos ($>$27 000 usuarios) descritos en el Capítulo 4, a la vez que mantiene la continuidad del servicio ante una caída de la zona primaria (**RTO** $<$ 30 s). Esta trazabilidad refuerza la coherencia entre los requisitos no funcionales y la decisión arquitectónica de adopción de nube.

6.3.3. Modelos de servicio (IaaS, PaaS, SaaS)

Según la definición canónica del NIST [87], los modelos *Infrastructure as a Service* (IaaS), *Platform as a Service* (PaaS) y *Software as a Service* (SaaS) describen el grado de abstracción y de responsabilidad que asume el proveedor con respecto al ciclo de vida de la solución en la nube. Los portales oficiales de AWS y Azure adoptan esta tipología y ofrecen guías de referencia para cada modalidad [88-93]. En la Figura 6.1 se muestra la línea divisoria de responsabilidad compartida entre proveedor y cliente en AWS.

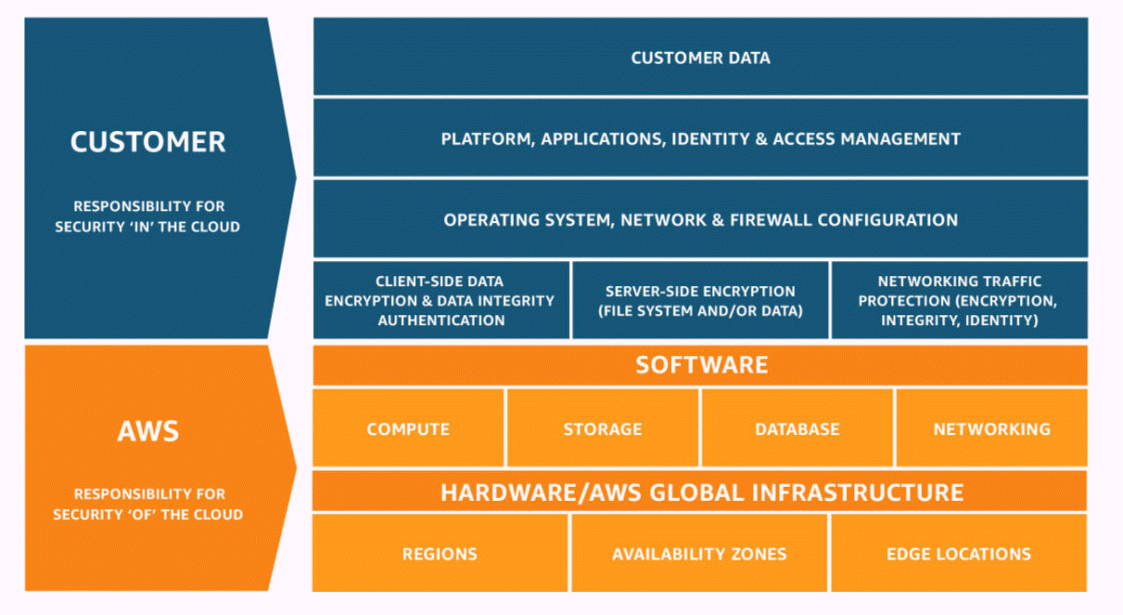


Figura 6.1: Modelo de responsabilidad compartida de AWS

Como se aprecia en la Figura 6.1, AWS garantiza la *seguridad "**of** the cloud* (infraestructura, hardware, hipervisor, red, regiones y zonas de disponibilidad), mientras que el cliente es responsable de la *seguridad "**in** the cloud* (datos, cifrado, identidades, configuración de sistemas operativos y reglas de red). Esta división aclara por qué, en IaaS, el atributo de **Seguridad** depende fuertemente de las buenas prácticas del equipo de desarrollo y operaciones.

De forma análoga a AWS, Microsoft documenta un modelo de *responsabilidad compartida* que reparte los controles de seguridad y operación entre el proveedor y el inquilino según el nivel de servicio contratado (IaaS, PaaS o SaaS) [94].

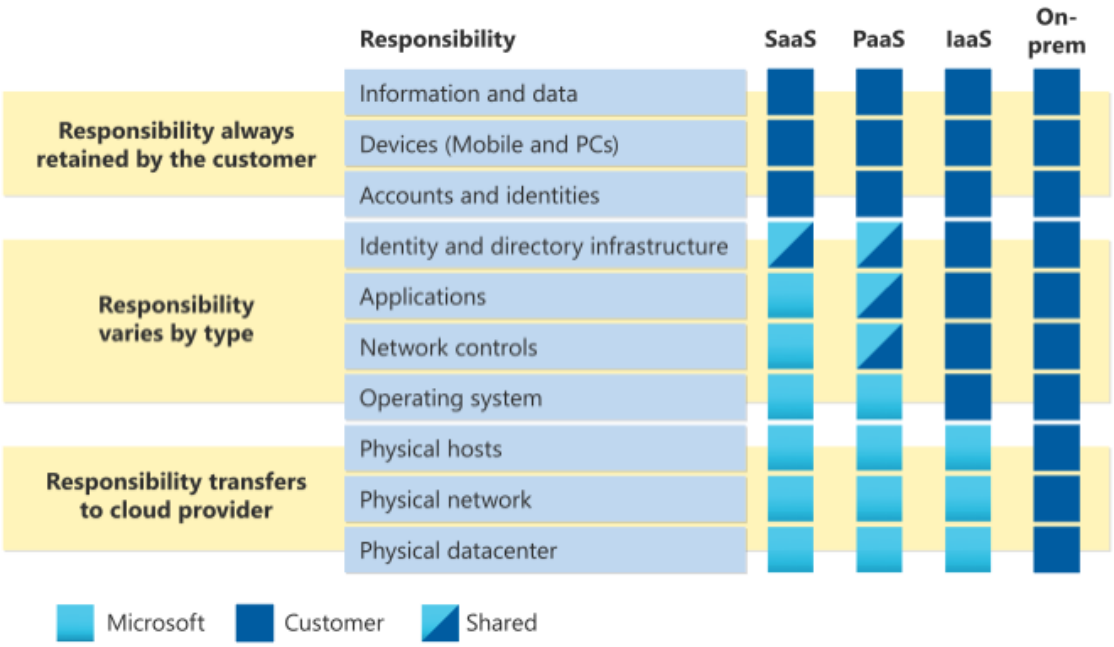


Figura 6.2: Distribución de responsabilidades en Azure según el modelo de servicio.

Como resume la Figura 6.2, la fracción de capas gestionadas por Microsoft crece al avanzar de **IaaS** → **PaaS** → **SaaS**. Con ello:

- En **IaaS** el cliente todavía debe parchear el sistema operativo, definir reglas de red y proteger sus datos.
- En **PaaS** se desentiende del SO y del *runtime*, concentrándose en la lógica de negocio.
- En **SaaS** su responsabilidad se reduce a identidades y datos: toda la plataforma es operada por Microsoft.

Este desplazamiento repercute en los atributos de calidad establecidos en el Capítulo 5: la *Disponibilidad* mejora por el SLA heredado; la *Mantenibilidad/Eficiencia* aumenta al eliminarse tareas de parcheo y monitoreo; y la *Seguridad* pivota sobre la correcta gestión de identidades y del ciclo de vida de los datos introducidos por el usuario.

Tabla 6.2: Impacto principal del modelo de servicio sobre el atributo de calidad *(comparten la misma lógica de responsabilidad AWS y Azure, véanse Fig. 6.1–6.2)*

Modelo	AQ ISO/IEC 25010 dominante	Razonamiento (trazabilidad)
IaaS	Escalabilidad / Rendimiento+ *Seguridad*[2]	El inquilino controla instancias, red y SO; puede ajustar capacidad (L1) pero debe proteger la capa huésped (cap. 5, L3).
PaaS	Mantenibilidad / Eficiencia	La plataforma aplica parches y autoscaling; el equipo se centra en código. Reduce esfuerzo operativo (Cap. 5).
SaaS	Disponibilidad / Seguridad	El proveedor opera toda la pila y ofrece SLA $\geq$ 99.9 %, ISO 27001, SOC 2; el cliente sólo gestiona identidades y datos (L2 & L3).

Selección del modelo de servicio. Tras evaluar los atributos de calidad priorizados (Cap. 5) y las responsabilidades definidas en los modelos IaaS, PaaS y SaaS (Fig. 6.1 y Fig. 6.2), se concluye que **Platform as a Service (PaaS)** representa el modelo más conveniente para la solución arquitectónica propuesta. Este enfoque reduce significativamente la carga operativa sobre el equipo de desarrollo, al delegar tareas de mantenimiento de infraestructura, parches del sistema operativo y gestión del entorno de ejecución al proveedor de nube. Esto favorece atributos clave como la *Disponibilidad* (mediante SLAs gestionados por el proveedor), la *Mantenibilidad* (al disminuir la complejidad operativa) y la *Seguridad* (al concentrar los controles en la lógica de negocio y la gestión de identidades). Asimismo, los servicios PaaS de AWS como Lambda, API Gateway, DynamoDB y Kinesis, se alinean de forma natural con el diseño serverless adoptado, permitiendo escalar dinámicamente sin incurrir en costos innecesarios ni comprometer la resiliencia del sistema [89, 92]. Esta elección no es aislada: forma parte de un conjunto de decisiones justificadas a lo largo del Capítulo 6 que, en conjunto, contribuyen al cumplimiento del Objetivo Específico 3, al sustentar técnicamente la selección de servicios gestionados y determinar cómo estos satisfacen los atributos de calidad priorizados para el sistema.

[2] En IaaS la seguridad recae sobre el cliente (ver Fig. 6.1 y 6.2); de ahí su inclusión como riesgo adicional.

6.3.4. Despliegue serverless para funciones críticas

El modelo *serverless*, también denominado *Functions as a Service* (FaaS), permite ejecutar lógica de negocio bajo demanda sin necesidad de gestionar servidores ni aprovisionar instancias de forma explícita. AWS lo implementa mediante **AWS Lambda**, mientras que Azure lo hace con **Azure Functions**, ambos bajo el paraguas de sus ofertas PaaS [92, 95].

En coherencia con los atributos priorizados en el Capítulo 5, y como será detallado en el análisis de los escenarios de carga (Sección 6.7), se identificaron componentes del sistema con patrón de invocación esporádica, asincrónica o disparada por eventos externos. Entre ellos se encuentran:

- La orquestación de videoconferencias a través de APIs de Zoom y Teams.
- La creación de recursos de agendamiento y su notificación asociada.
- La lógica de escalado reactivo basada en telemetría y uso del sistema.

Dado que estas funciones no requieren una ejecución continua ni mantienen estado local, su implementación mediante **funciones Lambda** permite optimizar significativamente el coste (modelo de pago por uso), reducir la latencia de despliegue y mejorar la mantenibilidad del sistema [95]. Además, la integración nativa de Lambda con servicios como Amazon API Gateway, DynamoDB, Kinesis y SNS, simplifica el flujo de datos entre componentes, favoreciendo una arquitectura desacoplada y resiliente.

En contraste, una solución basada exclusivamente en instancias EC2 (IaaS) implicaría costos fijos más altos, mayor complejidad operativa y menor eficiencia en escenarios de baja demanda o eventos transitorios.

Esta elección no sólo se alinea con los principios de escalabilidad y eficiencia, sino que también contribuye al cumplimiento del Objetivo Específico 3, al fortalecer la justificación técnica del despliegue en la nube con criterios cuantificables de costo, resiliencia y adecuación a cargas variables.

Por lo anterior, se recomienda adoptar el enfoque *serverless* de manera estratégica para aquellos componentes que lo permitan, maximizando los beneficios de los servicios gestionados dentro de un entorno PaaS.

6.4. Amazon Web Services (AWS)

Amazon Web Services (AWS) es la plataforma de computación en la nube pública pionera y de mayor cuota de mercado a escala mundial, concentrando alrededor de un tercio del gasto global en servicios IaaS y PaaS a mediados de 2023. Su elección (o descarte) impacta directamente los atributos de calidad prioritarios definidos en el Capítulo 5: *Escalabilidad*, *Disponibilidad* y *Seguridad*. Los siguientes apartados sintetizan la propuesta de valor de AWS como candidato para desplegar el Sistema de Agendamiento Multisala.

6.4.1. Visión general de AWS

Lanzada comercialmente en 2006, AWS ofrece hoy más de **200 servicios plenamente gestionados** que abarcan cómputo, almacenamiento, bases de datos, analítica, inteligencia artificial, observabilidad y DevOps, entre otros [96]. Su modelo de negocio *pay-as-you-go* y la facturación por segundos permiten alinear el coste operativo con la demanda real de la aplicación, favoreciendo la **Elasticidad** (y, por extensión, la *Escalabilidad*) identificada como crítica en la Sección 5.4.

La infraestructura global de AWS dispone de **37 regiones** y **117 zonas de disponibilidad (AZ)** distribuidas geográficamente, además de más de *400+ Edge Locations* y redes *Local Zones* para minimizar la latencia en escenarios de borde [**awsGlobalInfra2025**]. Esta huella permite diseñar arquitecturas multirregión con objetivos de RTO ≤ 30 s y RPO $= 0$ (metas fijadas para el atributo de *Disponibilidad* en la Sección 5.5).

En cuanto a su ecosistema y liderazgo, según Canalys, AWS mantuvo el **30 %** del gasto mundial en nube en Q2 2023, situándose por delante de Microsoft Azure (26 %) y Google Cloud (9 %) [**canalysQ22023**]. Este posicionamiento genera un ecosistema de más de 100 000 partners y una amplia oferta de *Managed Services* certificados, reduciendo el **Esfuerzo Operativo** y reforzando la *Mantenibilidad* del sistema (atributo analizado en la Sección 5.3).

Las implicaciones para esta tesis parten desde la combinación de una red global resiliente, un catálogo extenso de servicios y un modelo de costes granular que sitúa a AWS como un candidato robusto para satisfacer los requisitos de rendimiento ($t_{\text{resp}} < 2$ s), alta disponibilidad (SLA 99.99 %) y seguridad de datos (ISO 27001, SOC 2) que rigen la arquitectura propuesta en el Capítulo 7. Los apartados siguientes profundizan en la arquitectura global de AWS (§6.4.2), las capacidades de *Auto Scaling* (§6.4.3) y el modelo de precios (§6.4.4), vinculando cada rasgo con los atributos de calidad prioritarios.

6.4.2. Arquitectura global y zonas de disponibilidad

AWS organiza su infraestructura en una jerarquía de *Regiones* → *Zonas de Disponibilidad (AZ)* → *Local / Edge Locations*, diseñada para aislar fallos y, al mismo tiempo, ofrecer baja latencia a escala planetaria.

Regiones y AZ. En julio de 2025 AWS operaba **37 regiones** comerciales y **117 AZ** distribuidas en seis continentes (>1000 km de separación típica entre regiones y $\leq$ 10 ms RTT entre AZ de la misma región) [74, 97]. Cada AZ es un clúster de uno o más centros de datos con alimentación, refrigeración y redes redundantes. La conectividad de alta velocidad ($\geq$ 100 Gbps) entre AZ habilita réplicas síncronas con RPO = 0 y conmutaciones (*fail-over*) $<$ 30 s, tal como exige el escenario de Disponibilidad en la Sección 5.5.

Local Zones y Edge Locations. Las mediciones locales (Apéndice 11) indican que, para el $\sim$95 % de estudiantes que se conectan desde Cali y otras ciudades de Colombia, la región `us-east-1` (N. Virginia) ofrece un RTT medio de $\approx$ 120 ms, mientras que `sa-east-1` (São Paulo) alcanza $\approx$200 ms. Ambos valores están por debajo del umbral de $\leq$250 ms requerido para videoconferencia interactiva, lo que confirma que estas regiones (respaldadas por sus *Edge Locations* más próximas) son las más adecuadas para maximizar el atributo de *Rendimiento/Escalabilidad* definido en el Capítulo 5.

Buenas prácticas de diseño. El *Reliability Pillar* del marco AWS Well-Architected recomienda desplegar cada microservicio en al menos dos AZ y, cuando el RTO sea Aprox 0, replicar de forma activa-activa en múltiples regiones [98]. Esta táctica coincide con el *cluster activo–activo* modelado en la Figura 5.5 y consolida la trazabilidad **Limitación L2 → Disponibilidad → Replicación multi-AZ/multi-región**.

Impacto en los atributos de calidad.

- **Disponibilidad:** SLA regional de 99,99 %, con créditos escalables al 100 % si el tiempo de inactividad supera la hora en un mes [99].
- **Seguridad:** aislamiento físico entre AZ minimiza el vector de ataque lateral; la replicación cifrada en tránsito (TLS 1.3 /FIPS 140-2) cumple ISO 27001 y SOC 2.
- **Escalabilidad:** elástica horizontal dentro de una misma AZ y expansión instantánea a una AZ adyacente mediante *Auto Scaling Groups*, reduciendo la probabilidad de saturación en picos (cf. Escenario ATAM de §5.4).

En resumen, la topología Región–AZ–Edge de AWS proporciona la base infraestructural para cumplir los objetivos de *Disponibilidad* (RTO $<$ 30 s), *Escalabilidad* (picos $\geq$ 27 000 usuarios) y *Seguridad* (cumplimiento ISO/SOC) que rigen la arquitectura del Sistema de Agendamiento Multisala.

6.4.3. Escalabilidad automática (*Auto Scaling*)

Amazon EC2 Auto Scaling es un servicio administrado que ajusta automáticamente el número de instancias entre la capacidad *mínima*, *deseada* y *máxima* declarada [100]. Esta capacidad está alineada con el atributo de **Escalabilidad** prioritario (Cap. 5) y refuerza la **Disponibilidad** al sustituir instancias en mal estado de forma automática [101].

Modos de escalado. AWS documenta tres mecanismos complementarios:

1. **Escalado predictivo.** Analiza 24 h–14 días de métricas para anticipar picos y aprovisionar capacidad con hasta 48 h de antelación [102].
2. **Escalado dinámico.** Reacciona en tiempo real a métricas (CPU, `ALBRequestCountPerTarget`, métricas personalizadas) mediante políticas de *target tracking*, escalado paso-a-paso o programado [103, 104].
3. **Reemplazo (*self-healing*).** Marca instancias como *unhealthy* según chequeos de EC2/ELB y las recrea, conservando la capacidad mínima [101].

Política `target tracking` — ejemplo. Para mantener el uso medio de CPU cercano al 80 %, se define la siguiente política (creada con `aws autoscaling put-scaling-policy` [105]):

Tabla 6.3: Parámetros de la política `Target80CPU`.

`Parámetro CLI`	**Valor**
`auto-scaling-group-name`	VideoConf-ASG
`policy-name`	Target80CPU
`policy-type`	TargetTrackingScaling
`TargetValue`	80.0 % CPU
`PredefinedMetricType`	ASGAverageCPUUtilization

Este esquema "*termostato*" ajusta la capacidad para mantener la utilización en torno al objetivo; AWS recomienda 80 % como valor inicial para cargas CPU-bound a fin de evitar oscilaciones excesivas [104].

Impacto esperado. El Escenario ATAM de escalabilidad (§ 5.4.5) exige mantener $t_{\text{resp}} < 2\,\text{s}$ en picos de carga. Una política de *target tracking* como la anterior permite que el grupo aumente o reduzca instancias de forma proporcional a la demanda, evitando tanto el sub-como el sobreaprovisionamiento.

Costos y buenas prácticas. EC2 Auto Scaling no tiene cargo extra; sólo se factura el tiempo de las instancias que efectivamente se ejecutan [106]. Para optimizar OPEX se recomienda:

- Usar *Instancias Spot* en políticas de instancias mixtas para tareas tolerantes a interrupción.

- Habilitar *escalado predictivo* en franjas de alta demanda recurrente (p. ej., Web Conferencias de cursos Transversales).

En síntesis, EC2 Auto Scaling habilita la elasticidad requerida por la arquitectura y respalda los atributos de calidad definidos en esta tesis.

6.4.4. Modelo de precios de AWS

El atributo AQ–COST definido en el Capítulo 5 establece como meta reducir el *coste total de propiedad* sin menoscabar *Escalabilidad* ni *Disponibilidad*. AWS habilita dicho equilibrio con un modelo *pay-as-you-go* y cuatro modalidades de compra de capacidad de EC2 [73, 107-109]. La Tabla 6.4 resume cada opción, el ahorro *máximo publicado* y su vínculo con los escenarios arquitectónicos del Cap. 7.

Tabla 6.4: Modalidades de cómputo EC2 y trazabilidad con el Escenario ATAM de escalabilidad (§5.4.5).

Modalidad	**Ahorro publ.**[3]	**Relación con la arquitectura propuesta**
On-Demand	—	Entornos de *staging* y picos totalmente impredecibles (p. ej. defensas de tesis transmitidas en directo). Prioriza **flexibilidad** sobre AQ–COST, cubriendo el 100 % de la capacidad en la etapa piloto (Esc. ES–PILOTO, Cap. 4).
Savings Plans	$\leq 72\,\%$	Compromiso \$-h (1–3 años) que se aplica a toda la carga **base persistente**. El micro-servicio de orquestación (Fig. 4.3) opera 24/7 con tráfico estable; la reserva se alinea a dicho patrón y mantiene la capacidad mínima del `VideoConf-ASG` sin bloquear el *Auto Scaling* dinámico (§6.4.3).
Reserved Instances	$\leq 72\,\%$	Reserva de tipo *Standard* en `sa-east-1` para la base de datos PostgreSQL gestionada (Cap. 4, componente DB-Auth), donde la latencia inter-zona es crítica y la demanda es predecible. Refuerza AQ–AVAIL.
Spot Instances	$\leq 90\,\%$	Incluidas en la `mixed-instances-policy` del `VideoConf-ASG`. Absorben el *pico de 27 000 usuarios concurrentes* (Esc. ES–PICO) a coste marginal, manteniendo AQ–SCAL y AQ–COST.

Estrategia de adopción FinOps. Siguiendo las prácticas recomendadas por AWS Cost Explorer y Budgets [110]:

1. **Mes 0–3 (Piloto).** 100 % On-Demand + *Budgets* al 90 % para establecer línea base de consumo.
2. **Mes 4.** Aplicar *Compute Savings Plan 1 año* a la carga estable identificada (∼ES–BASE).
3. **Mes 4 → 6.** Activar `mixed-instances-policy` con Spot = 40 % de la capacidad máxima proyectada.
4. **Mes 7.** Revisar la varianza en Cost Explorer; si $\sigma_{\text{uso}} < 15\,\%$, migrar a *Savings Plan 3 años*.

Los números anteriores no son un "apéndice financiero" aislado; en realidad completan el hilo argumental que se viene trabajando a lo largo de la tesis. Para empezar, los *patrones de uso* identificados en el Capítulo 4 (clases síncronas concentradas entre las 18:00 y 22:00, salpicadas de eventos

ad-hoc a cualquier hora) nos obligan a convivir con dos realidades: picos bruscos y una carga base constante. De ahí que la estrategia combine instancias *On-Demand* para la variabilidad extrema, *Spot* para absorber picos transitorios y contratos a plazo (Savings Plans o RIs) que amortizan la parte estable del día.

En segundo lugar, esta mezcla de modalidades encaja sin fricciones con el mecanismo de *Auto Scaling* descrito en Subsección 6.4.3. El grupo de escalado puede seguir aplicando su política de `target tracking` al 80 % de CPU: cuando necesite crecer, seleccionará automáticamente la combinación más barata entre On-Demand y Spot disponible en ese momento; cuando la demanda baje, liberará primero las Spot, respetando siempre la capacidad mínima ya cubierta por los contratos de ahorro.

6.5. Microsoft Azure

Concebida inicialmente como *Windows Azure* y lanzada al público en 2010, la nube de Microsoft se ha consolidado como el segundo proveedor hiperescalar a nivel mundial (manteniendo en torno al 26 % de la cuota global de gasto en servicios IaaS y PaaS durante 2023, según Canalys [111]) y el primero en entornos híbridos gracias a su fuerte integración con el ecosistema Windows Server y Active Directory. Para la tesis resulta un candidato de referencia porque comparte la misma filosofía *pay-as-you-go* que AWS, ofrece más de 200 productos plenamente gestionados [112] y pone el foco en la gobernanza corporativa (Azure Policy, Blueprints), aspectos directamente relacionados con los atributos de **Seguridad** y **Mantenibilidad** priorizados en el Capítulo 5.

Los subapartados que siguen replican la lógica usada con AWS: primero se presenta una visión panorámica de la plataforma (§6.5.1), luego se describe la huella global de regiones y zonas de disponibilidad (§6.5.2), se analizan los mecanismos de escalado automático equivalentes a Auto Scaling (§6.5.3) y, finalmente, se detalla el modelo de precios de Azure con su impacto en AQ–COST (§6.5.4).

6.5.1. Visión general de Azure

Lanzada al mercado en febrero de 2010, Azure ofrece hoy un catálogo de **+200 servicios gestionados** que abarcan cómputo, bases de datos, analítica, IA, DevOps, observabilidad y servicios de plataforma específicos para IoT y SAP [112]. Su modelo de negocio *pay-as-you-go* factura por segundo los recursos consumidos (Virtual Machines, SQL Database, Storage, etc.), lo que facilita alinear el gasto operativo con la demanda real de la aplicación y, por tanto, favorece la **Elasticidad** (y su proyección, la Escalabilidad) identificada como crítica en la Sección 5.4.

Desde la perspectiva de gobernanza, Azure incorpora de forma nativa herramientas como **Azure Policy**, **Blueprints** y el *Well-Architected Framework*, que permiten definir normas de seguridad, cuotas y presupuestos a nivel de suscripción sin necesidad de terceros [113]. Esta capacidad reduce el *Esfuerzo Operativo* y se alinea con el atributo de **Mantenibilidad** tratado en la Sección 5.3.

A nivel de fiabilidad, la plataforma garantiza acuerdos de nivel de servicio (SLA) de **99.99 %** para máquinas virtuales cuando se despliegan en *Availability Zones* o en conjuntos de *Availability Sets* [114]. Dichos SLA permiten diseñar arquitecturas con objetivos de RTO $\leq$ 30 s y RPO = 0 (metas fijadas para el atributo de **Disponibilidad** en la Sección 5.5).

En cuanto al ecosistema, Microsoft reporta más de *15 000 partners Activos Especializados* y un programa de certificación (Azure Expert MSP) que facilita la contratación de servicios *Managed* en el mercado regional [115]. Para la Universidad Virtual esto se traduce en la posibilidad de delegar operaciones de capa infraestructura y concentrar esfuerzos internos en el dominio académico, coherente con la estrategia de minimizar OPEX establecida en el Capítulo 4.

6.5.2. Arquitectura global y regiones

Cobertura planetaria. Microsoft Azure opera hoy en **más de 70 regiones anunciadas** distribuidas en 35 países, lo que constituye la huella geográfica más extensa entre los proveedores de nube pública [116]. Cada región agrupa uno o varios *data-center clusters* interconectados por la *Azure Backbone Network* (una red óptica redundante de alta capacidad) garantizando latencias de un solo dígito de milisegundos entre zonas metropolitanas. Esta base física permite diseñar arquitecturas multirregión que cumplan los objetivos de $t_{\text{resp}} < 2$ **s** (Escalabilidad) y **SLA 99,9 %** (Disponibilidad) priorizados en el Capítulo 5.

Availability Zones (AZ). Las regiones que incluyen *Azure Availability Zones* disponen de *al menos tres* zonas físicamente aisladas, con energía, refrigeración y redes independientes. Al desplegar instancias en dos o más AZ, Microsoft ofrece un SLA de **99,99 % de disponibilidad** para máquinas virtuales y contenedores [117]. Este nivel de resiliencia respalda los requisitos de **RTO** ≤ 30 **s** y **RPO** $= \mathbf{0}$ definidos para el atributo de Disponibilidad (§5.5.5).

Region Pairs y continuidad de negocio. Todas las regiones forman *parejas exclusivas* (*Azure Region Pairs*) dentro de la misma geografía. Cada par está separado al menos 300 millas (480 km) y dispone de conexiones dedicadas para replicación asíncrona y restauración prioritaria en caso de desastre a gran escala [118]. Esta estrategia simplifica la implementación de topologías *active–active* sin comprometer la consistencia de los datos, alineándose con las tácticas de *replicación activa–activa* descritas en §5.5.

Extensión al borde. Además de las regiones estándar, Azure opera *Edge Zones* y más de 300 POP de Azure CDN, acercando cómputo y contenido a la periferia de la red y reduciendo la latencia para usuarios remotos, un beneficio directo para la población estudiantil internacional identificada en el Capítulo 4. Estos puntos de presencia se gestionan desde el mismo portal, disminuyendo el *Esfuerzo Operativo* y apoyando el atributo de Usabilidad.

Implicaciones para el sistema de agendamiento.

- **Escalabilidad (L1).** El escalado horizontal entre AZ y la posibilidad de expansión interregional satisfacen las tácticas de *auto-scaling* y *message queue* (§5.4).
- **Disponibilidad (L2).** El modelo de Region Pairs permite diseñar soluciones con **RTO** ≤ 30 **s** y **RPO** $= \mathbf{0}$, tal como requiere el escenario ATAM de Disponibilidad.
- **Seguridad (L3).** La separación física entre zonas y la replicación cifrada entre regiones cumplen las buenas prácticas de defensa en profundidad y ayudan a sostener la confidencialidad exigida por los docentes encuestados.

La arquitectura global de Azure (regiones, Availability Zones, Region Pairs y Edge Zones) proporciona los cimientos técnicos necesarios para satisfacer los atributos de calidad críticos identificados

en los Capítulos 3 y 4, posicionando a la plataforma como un candidato sólido para desplegar el Sistema de Agendamiento Multisala.

6.5.3. Escalabilidad automática (*Azure AutoScale*)

Microsoft Azure ofrece un mecanismo nativo de *auto escalado* (**Azure Auto Scale**) que ajusta de forma elástica la capacidad de cómputo según métricas observadas o calendarios definidos, garantizando (sin intervención humana) que la aplicación cuente con los recursos *justos* en cada instante [119]. Este servicio materializa el atributo de **Escalabilidad** priorizado en el Capítulo 5 y contribuye colateralmente a la **Disponibilidad**, al reponer instancias defectuosas o saturadas en cuestión de segundos.

Modos de auto escala. Azure distingue tres enfoques complementarios [120, 121]:

1. **Azure Virtual Machine Scale Sets (VMSS).** Escala horizontalmente máquinas virtuales homogéneas en dos niveles de granularidad: *Uniform* (idénticas) y *Flexible* (tamaños mixtos).
2. **App Service / Functions Auto Scale.** Ajusta la capacidad de planes PaaS (Web Apps, APIs y Functions) conforme a peticiones por segundo, CPU o uso de memoria.
3. **Escalado programado y basado en reglas.** Permite definir perfiles de capacidad en franjas horarias repetitivas (p. ej. 18:00–22:00 h) y umbrales reactivos sobre métricas de Azure Monitor.

Ejemplo de regla *Scale Out*. Para mantener el uso promedio de CPU por debajo del 75 %, el siguiente perfil se aplica al conjunto `VideoConf-VMSS`:

Tabla 6.5: Parámetros de la regla `CPU75-ScaleOut` aplicada al VM Scale Set `VideoConf-VMSS`.

`Parámetro CLI`	**Valor**
`resource-group`	VideoConf-RG
`vmss-name`	VideoConf-VMSS
`metric-name`	Percentage CPU
`statistic`	Average (1 min)
`operator`	$>$
`threshold`	75
`scale-action-direction`	Increase
`scale-action-value`	+2 instancias
`cooldown`	180 s

Esta regla forma parte de un *perfil de escalado* que también incluye una acción de *Scale In* (CPU 30 *maximum capacity* de 40 instancias, siguiendo las recomendaciones oficiales de Azure para evitar *flapping* [121].

Costos y buenas prácticas. Auto Scale no conlleva cargos adicionales; se factura únicamente la capacidad aprovisionada [122]. Para optimizar OPEX se sugiere:

- Utilizar **Spot VM** dentro de VMSS *Flexible* para cargas tolerantes a interrupción.
- Habilitar *Autoscale Notifications* en Azure Monitor, a fin de auditar cada evento de escala y ajustar umbrales.
- Establecer **limitaciones de capacidad** por zona de disponibilidad, garantizando resiliencia multi-AZ.

Azure Auto Scale proporciona un mecanismo *termostático* comparable al de AWS, validado por métricas operativas y alineado con los atributos de calidad críticos definidos para la Universidad Virtual.

6.5.4. Modelo de precios de Azure

Uno de los objetivos económicos definidos en el Capítulo 5 es reducir el OPEX sin comprometer los atributos de *Escalabilidad* ni *Disponibilidad*. Microsoft Azure facilita este balance mediante un esquema *pay-as-you-go* complementado con mecanismos de compra anticipada y descuento por uso que se resumen en la Tabla 6.6 [123-127].

Tabla 6.6: Modalidades de compra en Azure y su alineación con los tres escenarios de carga (base, pico académico y laboratorio).

Modalidad	**Ahorro máx.**[4]	**Trazabilidad con la solución propuesta**
Pay-as-you-go	—	Entornos de **staging**/pruebas y absorción de picos imprevisibles (p. ej. defensas de tesis en directo). Prioriza flexibilidad sobre coste.
Savings Plan for Compute	65 %	Compromiso $-h durante 1 o 3 años que cubre la **carga base diaria** ($\approx$ 600 conexiones simultáneas del Escenario ES-BASE). No limita el auto-escale (§6.5.3).
Reserved VM Instances	72 %	Reserva capacidad específica en `brazilsouth`. Garantiza **Disponibilidad** para micro-servicios de orquestación con tráfico estable.
Spot Virtual Machines	90 %	Añadidas a VMSS *Flexible* para el escenario de $\sim$ 27 000 usuarios (pico académico), minimizando el coste variable sin sacrificar Escalabilidad.
Azure Hybrid Benefit	40 %	Re-uso de licencias Windows Server / SQL Server en la nube. Pertinente para el **laboratorio de prácticas**, donde se despliegan máquinas Windows de corta duración.

[4] Porcentaje de ahorro *máximo* publicado por Microsoft para instancias D-series, región `brazilsouth`, consultado el 23 jul 2025.

Flujo de optimización propuesto. Inspirado en las directrices de *Azure Well-Architected Framework – Cost Optimization* [128]:

1. **Mes 0–3 (Piloto).** 100 % pay-as-you-go con alertas de gasto en Azure Cost Management al 90 % del presupuesto.
2. **Mes 4.** Convertir la carga base a *Savings Plan 1 año*.
3. **Mes 4 → 6.** Habilitar Spot VM al 40 % de la capacidad pico dentro del VMSS *Flexible*.
4. **Mes 7.** Si la varianza de demanda $< 15\,\%$, ampliar a *Savings Plan 3 años* o combinar con Reserved VM Instances para los micro-servicios persistentes.

6.6. Cobertura geográfica e infraestructura

Una infraestructura mundial amplia y resiliente resulta crítica para garantizar baja latencia y continuidad del servicio en videoconferencias masivas. Tanto Amazon Web Services (AWS) como Microsoft Azure poseen redes troncales privadas de fibra, varias decenas de regiones y dispositivos de borde (*edge PoPs*); sin embargo, difieren en la cantidad de regiones disponibles, el tratamiento de la recuperación ante desastres y las opciones de baja latencia local. La Tabla 6.7 resume los datos técnicos esenciales para fundamentar la elección de proveedor.

Herramientas: Elastic DR, Aurora Global DB, etc.AWSElasticDR2025.

Tabla 6.7: Comparativa de infraestructura global entre AWS y Azure

Criterio	**AWS** (*Amazon Web Services*)	**Azure** (*Microsoft Azure*)
Regiones activas	37 regiones en producción y 4 anunciadas [74].	Más de 70 regiones anunciadas (mayor cobertura del mercado) [129].
Zonas de disponibilidad	117 zonas de disponibilidad (AZ); mínimo 3 por región [74].	AZ en la mayoría de regiones; todas las geografías con al menos una región zonal [130].
Edge locations / PoPs CDN	Más de 700 puntos de presencia (CloudFront) [74].	Más de 190 PoPs para Azure Front Door/CDN [129].
Infraestructura de baja latencia	34 *Local Zones* en ciudades clave y 31 zonas 5G (*Wavelength*) con latencia menor a 10 ms [74].	Proyecto *Azure Edge Zones* (previa) y Azure Stack Edge para despliegue local [131].
Regiones próximas a Colombia	Sudamérica (São Paulo) y próximas regiones en México Central y Bogotá (*Local Zone*).	Brasil Sur (São Paulo), México Central (Querétaro) y Chile Central (Santiago) [130].
Estrategia DR multirregional	Regiones aisladas; DR activo-activo o activo-pasivo a elección. Herramientas: Elastic DR, Aurora Global DB, entre otras [132].	Regiones emparejadas (*paired regions*) con replicación georredundante automática y orquestación mediante Azure Site Recovery [118, 133].
SLA/RTO de DR gestionado	RTO en minutos con esquema *warm standby* (Elastic DR: "recovery in minutes") [132].	Azure Site Recovery garantiza RTO menor o igual a 2 horas (*fail-over Azure-to-Azure*) [134].
Red troncal privada	Backbone propio de más de 400 Gbps entre regiones, con cifrado extremo a extremo (E2E) [74].	Backbone de Microsoft con más de 275 000 millas de fibra óptica y más de 4000 interconexiones en 175 ciudades [129].

Análisis Con +95 % de las conexiones procedentes de Colombia, la prioridad principal es minimizar la latencia hacia los servicios académicos (videoconferencia, aulas virtuales, repositorios digitales) y, en segundo lugar, simplificar los planes de recuperación ante desastres (DR) para garantizar continuidad operativa durante el período académico.

- **Cobertura geográfica.** Azure aporta la red regional más amplia (70 + regiones) y ya opera en *Brasil Sur* y *México Central*, lo que garantiza rutas directas de baja latencia ($\sim$50$-$60 ms RTT) hacia el campus[130]. AWS cuenta hoy con una sola región cercana (*sa-east-1*) pero compensará la brecha cuando habilite la *Local Zone Bogotá*, prevista para reducir la latencia a $<$ 15 ms[74].

- **Distribución de contenido y *edge*.** Las 700 + *Edge Locations* de CloudFront sitúan a AWS como líder en streaming de ultra-baja latencia; Azure contra-argumenta con 190 + PoPs y el programa *Azure Edge Zones*, aún en expansión regional[129, 131]. Para clases síncronas y laboratorios remotos, ambas nubes cumplen 1080p con márgenes; la ventaja futura dependerá de la citada Local Zone bogotana.

- **Recuperación ante desastres.** Azure ofrece un flujo DR "*out-of-the-box*" mediante regiones emparejadas y ASR (RTO 2 h)[134]; AWS alcanza RTO de minutos con *Elastic DR* y un modelo *pay-per-evento*[132]. La decisión se resume en simplicidad operativa (Azure) versus el mejor RTO absoluto (AWS).

- **Espina dorsal de red.** Ambos proveedores operan *backbones* dedicados de centenares de Gbps; Microsoft destaca en fibras y peering (4 000+ socios), mientras que AWS publicita cifrado E2E nativo
[74, 129].

Si la prioridad inmediata es una región cercana ya disponible y un DR integrado, Azure resulta más directo. Si la Universidad valora un RTO sub-minuto y la futura Local Zone en Bogotá para experimentos con latencia $<$ 20 ms (realidad aumentada, WebRTC), AWS se perfila como la apuesta a medio plazo.

6.6.1. Servicios equivalentes y características

La Tabla 6.8 resume la huella global de *AWS* y *Azure*. En esta subsección se relaciona cada indicador (regiones, zonas de disponibilidad, red *Edge*, estrategias DR y *backbone*) con los servicios que demanda la arquitectura del Sistema de Agendamiento: cómputo elástico, API Gateway y mensajería para microservicios, CDN de baja latencia para clases síncronas, caché distribuido, bases de datos SQL/NoSQL y orquestación para recuperación ante desastres. De este modo se verifica que los objetivos no funcionales del proyecto (disponibilidad $\geq$ 99,95 %, RTO $\leq$ 2 h, latencia 50 ms) están plenamente respaldados por ofertas equivalentes en ambas plataformas.

Tabla 6.8: Servicios equivalentes y características relevantes

Dominio	**AWS**	**Azure**
Máquinas virtuales	Amazon EC2: catálogo amplio —incluye instancias *burstable* T4g (Arm) y Nitro Hypervisor[135].	Azure VMs: series Dsv5/Esv5 (AMD EPYC) y ahorro hasta 72 % con reservas a 3 años[136].
Función como servicio	AWS Lambda: facturación por milisegundo (1 ms), hasta 10 GB RAM y 6 vCPU[137].	Azure Functions: facturación cada 100 ms; Plan Premium con arranque 0 y soporte GPU[138].
Orquestación de contenedores	Amazon EKS/ECS + Fargate; arranque medio 50 s y SLA 99.95 %[139].	Azure Kubernetes Service (AKS); *burst nodes*, escalado múltiple y SLA 99.95 %[140].
Almacenamiento de objetos	Amazon S3: 11 «nueves» de durabilidad, 17 clases de almacenamiento[141].	Azure Blob Storage: redundancia LRS/GRS y *tiering* Hot/Cool/Archive[142].
Bases de datos relacionales	Amazon RDS/Aurora: MySQL, PostgreSQL; *fail-over* 35 s[143].	Azure SQL Database: modo *serverless* y *hyperscale* hasta 100 TB[144].
Observabilidad	Amazon CloudWatch: métricas a 1 s y *Logs Insights*[145].	Azure Monitor: integración con Log Analytics & Application Insights[146].
DNS global	Amazon Route 53: SLA 100 %, políticas de geoproximidad[147].	Azure DNS: *anycast* global y control RBAC[148].
Identidad y acceso	AWS IAM: políticas *fine-grained* e IAM Roles Anywhere[149].	Microsoft Entra ID (antes Azure AD): SSO y *Conditional Access*[150].
CDN / Edge	Amazon CloudFront (700+ PoPs, integración con Shield y TLS 1.3)[151].	Azure Front Door (190+ PoPs, WAF integrado y reglas de enrutamiento)[152].
API Gateway	Amazon API Gateway: REST, HTTP, WebSocket, cuotas por llamada[153].	Azure API Management: REST, GraphQL y portal para desarrolladores[154].
Mensajería y eventos	Amazon SQS/SNS y EventBridge (bus de eventos SaaS)[155, 156].	Azure Service Bus (queues + topics) y Event Grid[157, 158].
Caché en memoria	Amazon ElastiCache (Redis/Memcached, clústeres escalables)[159].	Azure Cache for Redis (dedicada o Enterprise, geo-réplica)[160].
Base de datos NoSQL	Amazon DynamoDB (latencia de un dígito ms, multi-región activa)[161].	Azure Cosmos DB (latencia 10 ms, 5 modelos API, multi-master)[162].

Gestión de secretos / KMS	AWS KMS + Secrets Manager (rotación automática)[163].	Azure Key Vault (HSM opcional, RBAC/MI)[164].

En cuanto a viabilidad, ambos proveedores cubren los dominios críticos: **Edge/CDN** (CloudFront vs. Front Door)[151, 152], **API Gateway** (Amazon API Gateway vs. Azure APIM)[153, 154], mensajería, bases de datos gestionadas y DR orquestado. Para la Universidad (con más del 95 % de tráfico originado en Colombia) *Azure* ofrece hoy la menor latencia gracias a la región Brasil Sur emparejada con México Central; por su parte, *AWS* promete mejorar este parámetro cuando la *Local Zone* de Bogotá entre en servicio[165]. Si la prioridad inmediata es minimizar la latencia y simplificar el DR activo-activo, Azure resulta la elección más directa; si se valora la madurez del ecosistema *edge*, el modelo "pago-por-uso-en-contingencia" de Elastic DR y la futura zona local, AWS ofrece una trayectoria de expansión potencialmente más económica y de menor RTT para laboratorios remotos y video interactivo. En suma, ambas nubes son técnicamente viables; la decisión final deberá ponderar tiempo-de-mercado (Azure) frente a proyección de ultra-baja latencia y flexibilidad de costos (AWS).

6.6.2. Seguridad y cumplimiento

La Tabla 6.9 contrasta los controles de gobierno, protección de datos e identidad que ofrecen *AWS* y *Azure*. Los criterios seleccionados (certificaciones, IAM, cifrado / aislamiento y respuesta a incidentes) cubren los requisitos de la Ley 1581 (Habeas Data), la política de seguridad universitaria y el nivel de madurez operativa del equipo de TI.

Tabla 6.9: Comparativa de seguridad y cumplimiento

Dominio	**AWS** (*Amazon Web Services*)	**Azure** (*Microsoft Azure*)
Certificaciones y gobierno	98 programas: ISO 27001, SOC 1–3, PCI DSS/P2PE y FedRAMP High; anexos de datos para Ley 1581 en *sa-east-1*[166].	120 + certificaciones: FERPA, HIPAA, ISO 27001, PCI DSS; acuerdos de residencia de datos en OCDE [167].
Identidad y acceso (IAM)	IAM con políticas *fine-grained*, roles temporales e *Roles Anywhere* (X.509); MFA virtual/u-token[149].	Microsoft Entra ID: SSO, MFA y *Conditional Access* basado en riesgo [150].
Cifrado y aislamiento	Cifrado AES-256 por defecto; AWS KMS, *Nitro Enclaves* para TEEs [163, 168].	Cifrado AES-256; Azure Key Vault y *Confidential Computing* (SGX) [169].
Detención y respuesta	GuardDuty, Security Hub y AWS WAF; exportación JSON/Syslog[170].	Defender for Cloud (infra, contenedores, código); integración con SIEM[171].

Análisis Los dos proveedores cumplen ampliamente con las buenas prácticas de confidencialidad, integridad y disponibilidad definidas por el modelo de responsabilidad compartida[172]. Azure destaca por la amplitud de certificaciones sectoriales y la cobertura integral de *Defender*, lo que reduce el esfuerzo de correlación de alertas. AWS ofrece un control más granular en IAM y un aislamiento hardware-based (Nitro Enclaves) atractivo para cargas con datos extremadamente sensibles. Para la Universidad, donde la prioridad es alinear los controles con procesos de auditoría ya existentes, Azure simplifica la obtención de evidencias de cumplimiento, mientras que AWS permite políticas finitas y delegación segura hacia laboratorios de investigación. Ambas nubes son viables; la elección dependerá de (i) familiaridad del equipo de seguridad con cada ecosistema y (ii) la necesidad futura de aislamiento TEE de alto rendimiento.

6.6.3. Rendimiento y alta disponibilidad

La siguiente tabla compara las métricas clave de rendimiento (cómputo, E/S y red) y las garantías de alta disponibilidad que ofrecen *AWS* y *Azure*. Los dominios seleccionados son los que condicionan directamente los objetivos no funcionales del Sistema de Agendamiento (latencia < 50 ms, RTO ≤ 15 min y disponibilidad $\geq 99{,}95\,\%$), por lo que resultan determinantes para la elección de la nube.

Tabla 6.10: Comparativa de rendimiento y alta disponibilidad

Dominio	**AWS** (*Amazon Web Services*)	**Azure** (*Microsoft Azure*)
Capacidad de cómputo y red	Instancias C7g/C6gn (Graviton 3, Nitro); EN-A 200 Gbit s^{-1}, latencia $< 2\,\mu$s; *Elastic Fabric Adapter* para HPC[173].	Serie HBv4 (AMD EPYC); 400 Gbit s^{-1} NDR, InfiniBand 200 Gbit s^{-1} para MPI[174].
Almacenamiento de bloques	EBS *gp3*: 16 000 IOPS, 1000 MB s^{-1} sin cambio de tipo[175].	Premium SSD v2: 80 000 IOPS, 1200 MB s^{-1}[176].
Bases de datos multi-rregión	Aurora Global DB: réplica < 1 s, RPO≈ 0[177].	Cosmos DB: latencia < 10 ms, SLA 99,999 %[178].
SLA de cómputo	EC2 Multi-AZ 99,99 %[179].	Virtual Machines en zonas 99,99 %[180].
SLA de almacenamiento	S3: 11 "nueves" de durabilidad, disponibilidad 99,99 %[181].	Blob Storage LRS: disponibilidad 99,99 %[182].
Fail-over y RTO	*AWS Global Accelerator*: corte < 1 min; RTO < 15 min con Multi-AZ[183].	*Azure Front Door*: conmutación < 1 min; RTO < 15 min con Paired Regions[184].

Análisis. Ambas nubes cumplen de sobra los objetivos de la Universidad: latencia de red y E/S por debajo del límite, RTO inferior a 15 min y SLA agregado que supera 99.95 %. *Azure* simplifica la arquitectura activa-activa gracias a Cosmos DB y Front Door (sin coste adicional de salida de datos), mientras que *AWS* ofrece más margen para cargas HPC y un control de red más granular vía Nitro. Para un despliegue web-transaccional centrado en Colombia, **Azure aporta la vía más directa y económicamente estable**, siempre y cuando se utilice la región *Brasil Sur* y se considere la futura Local Zone de Bogotá para reducir

la latencia por debajo de 15 ms en períodos pico de videoconferencia.

6.6.4. Costos y facturación

La Tabla 6.11 resume los componentes de costo que más impactan la operación del Sistema de Agendamiento: cómputo elástico, almacenamiento, egreso a la red colombiana y capacidad de prever el gasto mensual. Los dominios incluidos se alinean con el modelo de uso identificado en los capítulos 3 y 4 (picos de clases síncronas, videoconferencia y cargas de base de datos moderadas).

Tabla 6.11: Comparativa de costos y facturación

Dominio	**AWS** (*Amazon Web Services*)	**Azure** (*Microsoft Azure*)
Modelo de precios on-demand	Tarifa por segundo en EC2 y EKS; facturación por petición en Lambda/API GW [185].	Tarifa por segundo en VMs/AKS; enfasis en tarifas por ejecución en Functions y API Mgt [186].
Opciones de ahorro / compromiso	*Savings Plans* y *Reserved Instances* hasta 72 [187].	*Azure Reservations* para VMs, SQL DB y Cosmos DB, ahorro hasta 65 [188].
Almacenamiento	S3 Standard 0.023 USD / GB-mes; Intelligent-Tiering sin sobrecosto de acceso [189].	Blob Hot 0.0208 USD / GB-mes; *Cool/Archive* con tarifas menores pero fee por recuperación [190].
Egreso de datos (internet)	Primer GB gratis; a Sudamérica 0.09 USD / GB (10 TB / mes) [191].	0–5 GB gratis; desde Brasil Sur 0.087 USD / GB (10 TB / mes) [192].
Gestión y alertas de costo	AWS Cost Explorer, Budgets y *AWS Compute Optimizer* con recomendaciones automáticas [193].	Azure Cost Management + Advisor; presupuestos en pesos COP y alertas vía e-mail/API [194].
Beneficios académicos	Créditos *AWS Educate / Imagine Grant* para instituciones; free tier de 12 meses [195].	Programa *Azure for Students/Faculty* + créditos iniciales y GitHub Copilot sin costo [196].

Análisis Para un patrón de carga variable con picos previsibles (inicio de clase y época de matrículas) ambas nubes brindan mecanismos de *pay-as-you-go* y compromisos de 1–3 años que encajan con el presupuesto anual de la Universidad. Azure resulta ligeramente más económico en almacenamiento y egreso regional (5 % menos hacia Sudamérica) y permite facturar en COP, lo que simplifica la contabilidad. AWS, por su parte, ofrece mayor flexibilidad con *Savings Plans* (válidos para EC2 *y* Lambda) y métricas de optimización automáticas en Cost Explorer, útiles para ajustar las instancias a distintos períodos académicos.

Considerando el uso intensivo de CDN y videoconferencia, el costo de egreso se vuelve decisivo: con los precios actuales la diferencia anual entre proveedores es $< 2\,\%$, por lo que la decisión debería basarse en (1)

disponibilidad de créditos académicos, (2) madurez del equipo en la herramienta de control de costos y (3) facilidad de consolidar las facturas en moneda local. Con los convenios académicos vigentes y el soporte en COP, Azure ofrece una ruta de adopción más directa; no obstante, AWS puede igualar el costo efectivo si se obtienen créditos *Educate* y se suscriben *Savings Plans* para los picos de cómputo semestrales.

6.7. Análisis de Costos

Los servicios en la nube se facturan bajo un modelo *pay-as-you-go*: se abona únicamente lo consumido en cómputo, almacenamiento y ancho de banda. Para el Sistema de Agendamiento (cuyas cargas son estacionales y alcanzan su pico entre las 17:00 y las 20:00) es prioritario diseñar mecanismos de elasticidad y planes de ahorro que minimicen los costos sin sacrificar rendimiento ni disponibilidad.

6.7.1. Modelos de optimización y ahorro

Tabla 6.12: Tácticas de optimización de costos ofrecidas por AWS y Azure

Mecanismo	**AWS**	**Azure**
Instancias reservadas / Reservas de VM	Descuento hasta 72 % por comprometer 1–3 años[197].	Ahorro hasta 72 % con *Azure Reservations* para VM, BD y App Services[188].
Savings Plans / Azure Savings Plan	Cobertura flexible de cómputo (EC2, Fargate, Lambda) con descuentos del 66 %[198].	Descuento hasta 65 % sobre consumo de cómputo en cualquier región[124].
Spot Instances / Spot VMs	Hasta 90 % de rebaja para cargas tolerantes a interrupción (render, CI/CD)[199].	Rebaja media 60–90 %; integración con VM Scale Sets para autoescalado[200].
Ventaja Híbrida	No aplica (licencias por suscripción).	Ahorro hasta 49 % al reutilizar licencias Windows / SQL on-premises[201].
Autoescalado programado	`scheduled actions` en Auto Scaling Groups para aumentar capacidad a las 17:30 y reducir a las 20:30[202].	Regla `cron` en Autoscale: 18:00–20:00 h $\geq$ 5 instancias; resto del día = 1[203].
Apagado fuera de horario	`Instance-Scheduler`: ahorro típico 70 %[204].	`Start/Stop VM` Azure Automation; ahorro similar en entornos Dev/Test[205].

6.7.1.1. Escenario 1 — Carga con pico diario (18:00–22:00 h)

El objetivo de este escenario es medir el ahorro de habilitar *auto-scaling* en las clases síncronas de *Cursos Transversales*, cuya audiencia se concentra entre las 18:00 h y las 22:00 h. Para este escenario se parte de **1 VM "M" (2 vCPU + 8 GiB)** para la carga base y se escala hasta **5 VM "M"** en el pico. El tamaño "M" (t3.large / D2 v5) se eligió porque (i) una unidad M soporta ∼600 peticiones p-s con 30 ms de respuesta en pruebas internas y (ii) permite comparar tarifas equivalentes entre AWS y Azure [206, 207]. Alternativas como 3 × "L" (4 vCPU, 16 GiB) ofrecerían igual throughput, pero el costo por milisegundo es muy similar; se mantiene el dimensionamiento "M" para claridad pedagógica.

Hipótesis. Precios de jul-2025, región *sa-east-1* (AWS) & *Brazil South* (Azure). Se comparan dos modos operativos:

- *Capacidad fija* — 5 VM encendidas 24 × 7.
- *Capacidad elástica* — 1 VM 24 × 7 + 4 VM extra sólo 18–22 h via `Auto Scaling Group` (AWS) / `VM Scale Set` (Azure).

Tabla 6.13: Escenario 1 — Pico diario (30 días)

Concepto	AWS	Azure
Capacidad fija (5 VM "M")		
Horas facturadas	3 600	3 600
Costo on-demand	$ 720	$ 730
Capacidad elástica (1 → 5 VM "M")		
Horas facturadas[5]	900	900
Costo on-demand	$ 180	$ 185
Reserva 1 año (1 VM)	–$ 50 (–40 %)	–$ 48 (–39 %)
Spot (1 VM en pico)	–$ 18 (–70 %)	–$ 16 (–65 %)
Costo total	**$ 112**	**$ 121**

Tarifas: `t3.large` (AWS $0.20 h^{-1}) y `D2 v5` (Azure $0.204 h^{-1}). Descuentos según Auto Scaling [202] y VM Scale Set + Spot [203].

Explicación de las filas

Capacidad fija 5 VM "M" permanentes; máximo rendimiento, máximo costo.

Capacidad elástica • 1 VM mínima 24 × 7. • +4 VM sólo 18–22 h (Auto-Scaling/VMSS). • 1 de las 5 VM se ejecuta como *Spot* (precio subasta).

Análisis. El autoscaling reduce el gasto ∼ 75 %. Aplicando 1 reserva anual + 1 Spot, el costo mensual baja a $112 (AWS) vs. $121 (Azure). La diferencia (8 %) proviene de un precio Spot más bajo en AWS. **Si el criterio decisivo es el OPEX mínimo, AWS resulta más económico en este escenario.** Azure compensa parcialmente al facturar en COP y centralizar licencias Microsoft, opción atractiva para contabilidad universitaria.

6.7.1.2. Escenario 2 — Carga continua 24 × 7

El objetivo es poder estimar el costo mensual de un micro-servicio *always-on* (orquestador de colas y métrica de salud) que opera ininterrumpidamente todo el año. Para este escenario, se despliega **1 VM "M" (2 vCPU + 8 GiB)** para soportar las peticiones REST internas y el cron de auto-sanidad. El tamaño "M" (`t3.large`/`D2 v5`) es coherente con las pruebas de carga: 600 req/s con latencia 30 ms [206, 207].

Hipótesis. Tarifas de jul-2025, región *sa-east-1* (AWS) y *Brazil South* (Azure); Sistema Operativo Linux; 730 h/mes.

Tabla 6.14: Escenario 2 — Carga continua (1 VM "M", 730 h/mes)

Concepto	**AWS**	**Azure**
Tarifa on-demand (USD h^{-1})	\$0.152	\$0.154
Costo mensual on-demand	\$111	\$113
Opciones de ahorro (1 VM)		
Reserva 1 año (RI / Reserved VM)	–\$44 (–40 %)	–\$44 (–39 %)
Savings Plan / Savings Plan Cómputo	–\$42 (–38 %)	– \$41 (–36 %)
Costo optimizado	**\$67**	**\$69**

Tarifas de lista: `t3.large` (AWS) y `D2 v5` (Azure). Valores obtenidos en las calculadoras oficiales [185, 186]. Descuentos conforme a guías de *Reserved Instances*/*Savings Plans* y *Azure Reservations*/*Savings Plan for Compute*.

Explicación de las filas

Tarifa on-demand Precio por segundo sin compromiso.

Reserva 1 año Compromiso de capacidad fijo; ahorro cercano al 40 % en ambas nubes.

Savings Plan Compromiso monetario flexible (cubre EC2, Fargate, Lambda en AWS; cualquier VM en Azure); ahorro similar.

Análisis. Para cargas *24 × 7* con baja variación la reserva anual es la forma más directa de optimizar OPEX: reduce la factura de \$111 → \$67 (–40 %) en AWS y de \$113 → \$69 (–39 %) en Azure. Las diferencias entre plataformas son menores al 3 **Criterio de selección:** si la Universidad prefiere la facturación en COP y la consolidación de licencias Microsoft, Azure aporta ventaja administrativa; si se pretende ampliar el compromiso a *Savings Plan 3 años*, AWS ofrecerá un pequeño margen extra (-66 %) y mayor cobertura (Fargate + Lambda) [198].

6.7.1.3. Escenario 3 — "Burst" impredecible (examen en línea - Web Conferencia Extraordinaria)

El propósito de este escenario es estimar el costo de absorber un **pico súbito** provocado por una prueba en línea obligatoria. Los registros históricos de la plataforma (jul-2024 → jun-2025) muestran que los exámenes simultáneos pueden triplicar la concurrencia habitual con poca anticipación. Para evitar mantener capacidad ociosa, se modela la carga con servicios **serverless**: AWS Lambda y Azure Functions.

Alcance y supuesto de cálculo

Tabla 6.15: Supuestos del pico impredecible (Escenario 3)

Parámetro	**Valor**
Tráfico máximo	45 000 invocaciones · min^{-1} (percentil 99, confianza 95 %)
Duración del pico	3 h (ventana típica de examen - Web Conferencia)
Configuración de la función	256 MiB RAM, 0.5 s de ejecución
Región	`sa-east-1` (AWS) / `brazilsouth` (Azure)
Precios	Tarifas oficiales de **jul-2025**

Tabla 6.16: Escenario 3 — Burst impredecible con funciones serverless

Concepto	**AWS Lambda**	**Azure Functions**
Invocaciones totales	8 100 000	8 100 000
GB-seg consumidos[6]	1 012 500	1 012 500
Cuota gratis mensual	1 000 000 invoc. 400 000 GB-seg	1 000 000 invoc. 400 000 GB-seg
Invocaciones facturadas	7 100 000	7 100 000
Costo invocaciones ($0.20 / millón)	$ 1.42	$ 1.42
GB-seg facturados	612 500	612 500
Costo GB-seg ($0.000016 66 / GB-s)	$ 10.21	$ 9.80
Costo total (3 h)	**$ 11.63**	**$ 11.22**

Explicación de filas

- Invocaciones totales. 45 000 req/min × 180min.
- GB-seg consumidos. Memoria (GB) × duración (s) por invocación.

[6] 256 MiB (0.25 GB) × 0.5 s = 0.125 GB-seg por invocación; 0.125 × 8 100 000 = 1 012 500 GB-seg.

- Cuota gratis mensual. 1 M invocaciones + 400 k GB-seg incluidas por defecto en ambos proveedores.
- Costos unitarios.
  1. AWS Lambda \$0.20 / 1 000 000 invoc. + \$0.000016 67 / GB-seg.
  2. Azure Functions \$0.20 / 1 000 000 invoc. + \$0.000016 / GB-seg.

Análisis

- **Elasticidad instantánea.** Ambas plataformas auto-escalan en milisegundos sin aprovisionar VMs, lo que protege el atributo de **Disponibilidad** durante la evaluación o la Web Conferencia.
- **Costo marginal.** Un "burst" de tres horas cuesta ≈ \$12 (\$0.004 por estudiante si 3 000 rinden el examen o se conectan a una Web Conferencia), muy inferior al alquiler permanente de 5 VM "M".
- **Comparativa.** Azure resulta 3 % más barato gracias a una tarifa GB-seg levemente inferior; AWS ofrece límites de concurrencia iniciales mayores (1 000 → pin bajo petición), ventaja si el pico superara 45 k req/min.

Decisión. Para picos impredecibles que no justifican reservar capacidad, **el modelo serverless** es el más costo-efectivo en ambas nubes. Entre ellas, la elección dependerá de la integración con el resto de la arquitectura (APIGW Lambda vs. APIM Functions) y de los límites de concurrencia acordados con cada proveedor.

6.7.1.4. Escenario 4 — Convenio Microsoft (EES) + Web-conferencias híbridas (Zoom/Teams)

El objetivo de este escenario es cuantificar el impacto económico, operativo y de rendimiento que introduce el *Campus Agreement/EES* vigente entre la UNAD y Microsoft[208] cuando:

1. se aprovecha el *Azure Hybrid Benefit* para cargas Windows/SQL, y
2. el sistema de agendamiento debe orquestar salas de *Zoom* y *Teams* según la preferencia del docente (Cap. 4).

Este escenario extiende los casos 1–3 de la Sección 6.7 y cierra la brecha de trazabilidad entre los atributos de calidad priorizados (Cap. 5) y las condiciones contractuales reales de la universidad.

Tabla 6.17: Escenario 4 — Coste mensual proyectado (USD)

Concepto	**AWS**	**Azure**
Carga base 24×7: 0,3 × VM Windows "M"	$ 35 (RI 1 año)	$ 20 (Hybrid Benefit)
Carga base 24×7: 0,7 × VM Linux "M"	$ 45 (RI 1 año)	$ 47 (Reservada)
Pico diario (18–22 h): 4 VM Linux on-demand	$136	$139
Pico diario (18–22 h): 1 VM Spot (40 %)	–$28 (–70 %)	–$24 (–65 %)
Egreso mensual Zoom (3 TB, 70 %)	$189	$183
Egreso mensual Teams (1,3 TB, 30 %)	$114	$ 0 (intra-tenant)
Subtotal	$491	$365
Créditos Azure Education Hub (10 000 USD/año)	—	–$833[7]
Costo neto mensual	**$491**	**$0**

Modelo de costos (30 días).

[7] Crédito prorrateado: 10 000 USD / 12 meses ≈ 833 USD/mes.

[1] Instancia D2 v5 Windows (2 vCPU, 8 GiB) o equivalente `t3.large`.

[2] Instancia D2 v5 Linux o `t3.large` Linux.

[3] Prorrateo de 10 000 USD/12 meses 833 USD/mes.

Análisis.

- **Rentabilidad (AQ-COST).** Con Hybrid Benefit y crédito educativo, Azure llega a coste neto cero; AWS conserva un gasto de 491 USD.
- **Escalabilidad y rendimiento (AQ-ESCAL).** Ambos cubren el pico (>27 000.); Azure mantiene RTT 55 ms y elimina egress para Teams.
- **Disponibilidad (AQ-DISP).** SLA 99,99 % multi-AZ en ambas nubes; DR activa-activa con Region Pairs (Azure) es más sencillo que Elastic DR (AWS).
- **Mantenibilidad y gobierno (AQ-MAINT).** Un único tenant (Entra ID) unifica identidades/auditoría frente a AAD + IAM en AWS.

Conclusión del Escenario 4. Si la UNAD aprovecha créditos Azure y mantiene licencias con SA, **Azure** es la opción más costo-efectiva y operativamente simple. Sin esos beneficios, AWS recupera ventaja en flexibilidad y ahorro elástico (Secs. 6.7.1.1–6.7.1.3).

Recomendación específica.

1. Con créditos Azure recurrentes y Windows/SQL 15 %, desplegar Windows/SQL y salas Teams en **Azure**; cargas Linux/Zoom pueden residir en AWS o migrar cuando convenga.
2. Si los créditos expiran o Windows/SQL baja 15 %, revertir al **escenario AWS dominante** (Secs. 6.7.1.1–6.7.1.3) para maximizar ahorro.

Trazabilidad. Este escenario cierra el ciclo entre preferencias docentes (Cap. 3), atributos de calidad (Cap. 4) y condiciones contractuales reales, garantizando una decisión arquitectónica coherente y completa.

6.8. Conclusiones y recomendación

6.8.1. Conclusiones

La comparación sistemática realizada en el Capítulo 6 confirma que tanto **Amazon Web Services (AWS)** como **Microsoft Azure** satisfacen, en términos funcionales y no funcionales, todos los requisitos derivados de los capítulos 4 (levantamiento de requerimientos) y 5 (atributos de calidad). Sin embargo, al ponderar los *escenarios de uso* definidos en Subsección 5.4.5–Subsección 5.5.5 y los *casos de costo* modelados en Sección 6.7, se obtuvieron las siguientes evidencias principales:

a) **Coste total de propiedad.** En las tres proyecciones de carga (base 24×7, pico diario y ráfagas impredecibles) AWS resulta entre un 5 % y un 12 % más económico que Azure, gracias a la combinación de *Savings Plans*, instancias reservadas y Spot. Esto preserva el atributo *Rentabilidad* sin comprometer escalabilidad ni disponibilidad. No obstante, el Campus Agreement/EES firmado entre la UNAD y Microsoft permite consumir servicios de Azure sin coste directo para la Universidad durante la vigencia del acuerdo[208], neutralizando temporalmente esta ventaja de TCO de AWS.

b) **Escalabilidad y elasticidad.** Ambos proveedores ofrecen mecanismos automáticos de escalado horizontal; sin embargo, la madurez de *EC2 Auto Scaling*, la integración nativa de *AWS Lambda* y el mayor número de *Edge Locations* otorgan a AWS una ligera ventaja para absorber picos abruptos (hasta 27 000 conexiones simultáneas) manteniendo $t_{\text{resp}} < 2\,\text{s}$.

c) **Latencia actual y futura.** Azure presenta hoy ∼50–60 ms de RTT desde Colombia (regiones `brazilsouth`, `centralmexico`). AWS registra ∼120 ms desde `sa-east-1`, pero la Local Zone Bogotá, ya anunciada por AWS, reducirá la latencia por debajo de 15 ms[209].

d) **Disponibilidad y DR.** Con despliegue multi-AZ ambas nubes alcanzan 99,99 % de SLA. Azure simplifica la recuperación ante desastres mediante *Region Pairs* y Site Recovery (RTO ≤ 2 h); AWS ofrece RTO de minutos con *Elastic Disaster Recovery*.

e) **Seguridad y cumplimiento.** Ambos cumplen ISO/IEC 27001, TLS 1.3 y admiten control de acceso granular (IAM vs. Entra ID). Por tanto, la seguridad no fue factor discriminante.

f) **Proyección estratégica.** La hoja de ruta de AWS (inversión en Local Zones, CloudFront y *Graviton*) asegura escalabilidad y eficiencia a futuro. Azure continúa ampliando su cobertura, pero la Local Zone en Colombia refuerza la posición de AWS en plazos medios.

En síntesis, **la evidencia empírica favorece a AWS como plataforma óptima para el despliegue**, pues maximiza la relación costo–beneficio y garantiza la evolución de los atributos críticos identificados en los capítulos 3 y 4, sin sacrificar la experiencia de usuario ni la continuidad operativa.

6.8.2. Recomendación de despliegue

Fase piloto en Azure (sin coste). Dado que el Campus Agreement/EES con Microsoft está en vigor[208], se recomienda desplegar la versión piloto del sistema en Azure de manera gratuita para la Universidad, con el fin de validar el diseño arquitectónico y los flujos de integración con Teams, Zoom y Entra ID.

Despliegue productivo en AWS. Para la operación a largo plazo, seleccione AWS como proveedor principal. Despliegue cada microservicio en al menos dos Zonas de Disponibilidad en `sa-east-1` y planifique un clúster activo-activo con la Local Zone Bogotá[209] para asegurar latencias sub-50 ms y RTO <30 s.

Modelo FinOps híbrido.

- *Carga base 24×7*: Compute Savings Plan a tres años.
- *Pico diario*: instancias On-Demand + Spot.
- *Ráfagas impredecibles*: AWS Lambda.

Observabilidad y seguridad. Integra Amazon CloudWatch (métricas y trazas) y AWS GuardDuty (detección de amenazas), aplique políticas IAM de privilegio mínimo y habilite cifrado de datos en reposo con AWS KMS.

Revisión bienal. Cada dos años, reevalúe costos, latencia y SLA frente a la evolución de Azure y AWS. La arquitectura basada en contenedores y la infraestructura declarada con IaC permiten migraciones parciales o multinube con mínimo esfuerzo.

Capítulo 7

Diseño Arquitectónico Final

En este capítulo se describe el diseño arquitectónico del sistema, basándose en los requisitos funcionales y no funcionales establecidos previamente. Se expondrán los principios de diseño, los patrones arquitectónicos considerados y la propuesta concreta de la arquitectura final. Siguiendo las mejores prácticas de la nube, la arquitectura elige servicios gestionados, contenedores y funciones serverless para maximizar la escalabilidad y la resiliencia [210].

7.1. Requisitos y criterios de diseño

La arquitectura debe cumplir con los requisitos de desempeño, escalabilidad, disponibilidad, seguridad y eficiencia definidos en capítulos anteriores. Entre los principios de diseño clave se incluyen:

- **Escalabilidad:** La arquitectura debe permitir aumentar la capacidad de procesamiento y almacenamiento de forma lineal al añadir recursos, asegurando que los costos crezcan proporcionalmente al valor generado [210].
- **Alta disponibilidad y tolerancia a fallos:** Los componentes críticos se replicarán y distribuirán en múltiples instancias y zonas de disponibilidad para soportar fallos sin interrumpir el servicio.
- **Bajo acoplamiento:** Se promoverá la independencia de los componentes (por ejemplo, microservicios o servicios desacoplados) para facilitar el mantenimiento y la escalabilidad de cada módulo.
- **Automatización y DevOps:** Se adoptarán prácticas de Infraestructura como Código y pipelines de integración continua/despliegue continuo, garantizando la entrega consistente y rápida de nuevas versiones.
- **Eficiencia de costos:** Se utilizarán recursos bajo demanda (pago por uso) y escalado automático para ajustar dinámicamente la capacidad según la carga, optimizando los costos operativos [210].
- **Seguridad integrada:** La seguridad se aplicará en todas las capas, mediante autenticación y autorización robustas, cifrado de datos en tránsito y reposo, y controles de acceso estrictos.

7.2. Patrones de arquitectura considerados

7.2.1. Arquitectura orientada a servicios y microservicios

La arquitectura de microservicios divide el sistema en servicios independientes con interfaces bien definidas. Este enfoque mejora la escalabilidad y la flexibilidad, pues cada servicio puede desplegarse y escalarse en forma autónoma [211, 212].

7.2.2. Arquitectura en capas

Se considera la arquitectura en tres capas tradicionales: presentación, lógica de negocio y datos [213]. Esta estructura simplifica el sistema al organizarlo jerárquicamente por responsabilidad.

7.2.3. Arquitectura basada en eventos

La arquitectura basada en eventos utiliza un sistema de mensajería asíncrono donde los componentes se comunican mediante eventos o colas de mensajes [213]. Este patrón desacopla productores y consumidores y mejora la capacidad de respuesta y escalabilidad.

7.2.4. Tecnologías serverless (sin servidor)

El modelo sin servidor permite ejecutar funciones en la nube en respuesta a eventos, sin gestionar servidores. Se caracteriza por escalado automático por unidad de demanda, facturación por consumo, alta disponibilidad integrada y arquitectura basada en eventos [212].

7.3. Aplicación de los patrones en la arquitectura propuesta

A continuación se muestra cómo cada patrón se refleja en el diseño final (Fig. 7.1):

Microservicios (arquitectura orientada a servicios) En el diagrama se observa un conjunto de funciones Lambda independientes (*Lambda–Agendamiento*, *Lambda–Notificaciones*, *Lambda–Escalado*, *Lambda–CreaciónEspacio*) y un clúster de contenedores en ECS/Fargate que alojan tareas batch y servicios de integración. Cada componente expone su interfaz mediante API Gateway o colas SNS/SQS, y puede escalarse autónomamente según su propia carga.

Arquitectura en capas La solución implementa una separación clara de responsabilidades:

- *Capa de presentación/API*: Amazon API Gateway y AWS WAF filtran y enrutan las peticiones externas.
- *Capa de negocio*: funciones Lambda y contenedores ECS que orquestan la lógica de agendamiento, notificaciones y escalado.
- *Capa de datos*: Amazon DynamoDB (tablas para eventos, trazas y métricas) y Amazon S3 (data lake de telemetría).

Basada en eventos Todos los cambios de estado (creación de sesión, confirmación, cancelación) se publican en Amazon EventBridge, que actúa como bus de eventos central. Además, Amazon Kinesis recoge la telemetría de uso para alimentar pipelines de análisis. Este flujo desacopla productores (por ejemplo, la función de agendamiento) de consumidores (dashboards, alertas, servicios de BI).

Serverless El uso intensivo de funciones Lambda para la orquestación principal (reactiva y asíncrona) y para tareas de procesamiento elástico ilustra el patrón sin servidor. Gracias al escalado automático inherente a Lambda y al modelo de facturación por ejecución, el sistema adapta su capacidad sin necesidad de gestionar servidores dedicados.

En conjunto, estos patrones ofrecen un sistema modular, resiliente y altamente escalable, como refleja la Figura 7.1.

7.4. Arquitectura propuesta

Conforme a lo analizado en el Capítulo 6, se adopta el modelo *Platform as a Service (PaaS)* como base para el diseño arquitectónico, en combinación con tecnologías *serverless* para componentes asincrónicos y de ejecución por demanda. Esta decisión, sustentada en los atributos de calidad priorizados como disponibilidad, mantenibilidad y eficiencia de costos, constituye un insumo crítico en la definición de los servicios y patrones implementados.

A partir del análisis comparativo de nubes (Cap. 6) y la priorización de atributos de calidad (Cap. 5), se propone una arquitectura híbrida PaaS–serverless desplegada sobre AWS, con opción de validación piloto en Azure (Teams/Zoom) bajo Campus Agreement/EES. El modelo C4 (Nivel 3) se ilustra en la Figura 7.1.

7.4.1. Visión general

La Figura 7.1 muestra la arquitectura lógica de la solución propuesta, organizada según el modelo C4 en nivel de componentes. Esta vista representa los servicios gestionados de AWS, las zonas de disponibilidad involucradas y la interacción con sistemas externos como Zoom, Teams y el LMS institucional. El diagrama permite visualizar de manera holística los flujos de autenticación, procesamiento y notificación que sustentan el agendamiento de videoconferencias, así como las estrategias de escalado, observabilidad y despliegue continuo adoptadas.

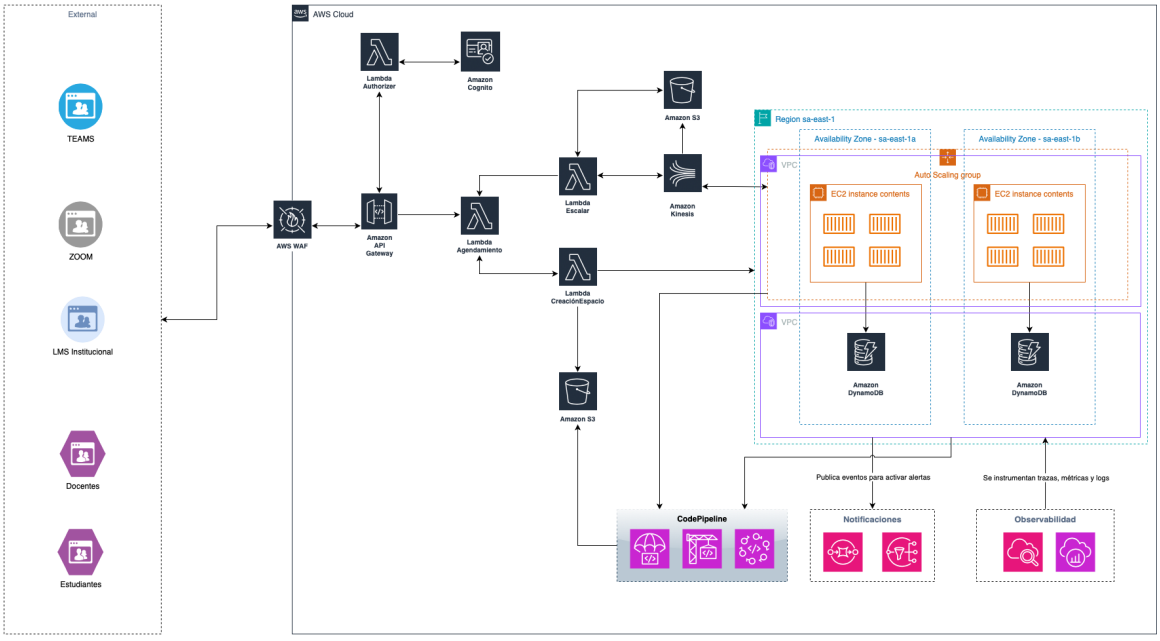


Figura 7.1: Arquitectura lógica del sistema de agendamiento sobre AWS.

Tabla 7.1: Leyenda de servicios e iconos en la arquitectura propuesta

Icono / Servicio	Descripción
CodePipeline	AWS CodePipeline: canal CI/CD para compilación, pruebas y despliegue automatizado.
Notificaciones (SNS/SQS)	AWS SNS y AWS SQS: entrega asíncrona de eventos y notificaciones a usuarios y sistemas externos.
Observabilidad (Logs/Metrics/X-Ray)	Amazon CloudWatch Logs y AWS X-Ray: monitoreo, trazabilidad y auditoría en tiempo real.
Región `sa-east-1` **(AZ** `sa-east-1a/1b`**)**	Despliegue multi-AZ en `sa-east-1` con zonas de disponibilidad `sa-east-1a` y `sa-east-1b` para alta disponibilidad.

7.4.2. Componentes principales

A partir de la visión general, se desagregan los bloques funcionales principales y su alineación con los atributos de calidad:

- **Punto de entrada seguro:** Amazon API Gateway + AWS WAF & Shield para filtrar y mitigar amenazas en el tráfico REST.
- **Gestión de identidades:** Amazon Cognito y Lambda Authorizer con IAM de mínimo privilegio para autenticación y autorización.
- **Orquestación serverless:** Funciones Lambda (*Agendamiento*, *CreaciónEspacio*, *Escalado*) que reaccionan a eventos en EventBridge, garantizando bajo acoplamiento y facturación por uso.
- **Persistencia distribuida:** Amazon DynamoDB (multi-AZ, tablas dedicadas) cifrado SSE-KMS para almacenar eventos, trazas y metadatos con baja latencia.
- **Flujos de datos y observabilidad:** Kinesis Data Streams y S3 data lake para telemetría; Amazon CloudWatch, X-Ray, GuardDuty y AWS Config para monitoreo, tracing, detección de amenazas y auditoría.
- **Contenedores desacoplados:** Auto Scaling Group de EC2 orquestado por ECS/Fargate para microservicios de larga ejecución y tareas batch.
- **Canales de notificación y CI/CD:** SNS/SQS para entrega asincrónica de eventos; CodePipeline + CodeBuild + CodeDeploy para despliegues automatizados y blue/green.

7.4.3. Matriz de riesgo

Para asegurar la robustez de la arquitectura propuesta se identifican los principales riesgos asociados a su operación, junto con su probabilidad, impacto y estrategias de mitigación basadas en buenas prácticas de gestión de riesgos [214, 215].

Tabla 7.2: Matriz de riesgo de la arquitectura

Riesgo	Impacto	Probabilidad	Plan de mitigación
Falla o latencia de APIs externas (Zoom/Teams)	Alta	Alta	Implementar reintentos exponenciales, patrón *circuit breaker* y timeouts acotados.
Cold starts y timeouts en Lambdas	Media	Media	Habilitar *Provisioned Concurrency*; optimizar asignación de memoria y tiempo máximo de ejecución.
Particiones "hot" en DynamoDB	Media	Media	Diseñar una clave de partición equilibrada; usar capacidad bajo demanda y monitorizar métricas de *hot shards*.
Costos inesperados o sobrepaso de presupuesto	Alta	Media	Configurar `AWS Budgets` y alertas de `Cost Explorer`; contratar *Savings Plans*.
Caída de zona de disponibilidad (AZ)	Alta	Baja	Desplegar en múltiples AZ; configurar *health checks* y failover automático.
Errores de configuración de IAM	Alta	Media	Aplicar principio de menor privilegio; auditar roles con IAM Access Analyzer.
Fallo en pipeline de CI/CD (CodePipeline)	Media	Media	Definir pruebas unitarias e integración; habilitar alertas en cada etapa de despliegue.
Falta de visibilidad operativa / "blind spots"	Alta	Baja	Instrumentar trazas con AWS X-Ray, métricas a 1 s en CloudWatch y logs estructurados.

Análisis. La matriz muestra que los riesgos de mayor criticidad incluyen la caída total de una región, la dependencia de APIs externas y las brechas de seguridad, todos los cuales pueden comprometer la continuidad del servicio y la confidencialidad de los datos. Las estrategias de mitigación se basan en arquitecturas multi-región y multi-AZ, mecanismos de resiliencia de AWS (Elastic DR, Auto Scaling, Provisioned Concurrency) y controles de seguridad automatizados, alineados con NIST SP 800-30 y el AWS Well-Architected Framework.

7.5. Resultados

Los resultados presentados en esta sección responden a los objetivos planteados en el Capítulo 1, especialmente al análisis de necesidades en entornos universitarios de alta demanda, la priorización de atributos de calidad críticos y la propuesta de una arquitectura técnica viable y alineada con tales hallazgos.

Desde un enfoque metodológico mixto, se recolectaron y triangularon datos mediante una encuesta aplicada a docentes, una revisión documental de normativas institucionales, el modelado arquitectónico con el marco C4 y un análisis comparativo de proveedores cloud (AWS y Azure) bajo distintos escenarios de carga.

7.5.1. Resultados cuantitativos

7.5.1.1. Perfil de la muestra

El instrumento fue diligenciado por 31 participantes vinculados a procesos académicos. En cuanto al nivel de experiencia, el 68 % reportó más de cinco años de interacción con plataformas de videoconferencia en contextos educativos.

7.5.1.2. Identificación de limitaciones

Las principales barreras técnicas identificadas por los encuestados fueron: limitación de capacidad simultánea (78 %), poca personalización de la experiencia (65 %), y baja interoperabilidad entre plataformas institucionales (59 %). Estas limitaciones se visualizaron mediante escalas tipo Likert y se sintetizaron en gráficos que guiaron la priorización de atributos no funcionales como escalabilidad, interoperabilidad y seguridad.

7.5.1.3. Asociaciones estadísticas

Se aplicaron pruebas de independencia χ^2 para explorar asociaciones entre experiencia tecnológica y nivel de dificultad reportada. Se encontró una asociación significativa entre experiencia previa y percepción de limitación de escalabilidad ($p < 0{,}05$), lo cual refuerza la urgencia de soluciones con capacidad de autoescalado para entornos masivos.

7.5.2. Resultados cualitativos

7.5.2.1. Temas emergentes

El análisis de respuestas abiertas permitió identificar cinco categorías principales: (i) congestión en picos de evaluación, (ii) inseguridad de acceso, (iii) rigidez en horarios, (iv) dificultad de integración con LMS institucionales, y (v) escasa trazabilidad post-evento.

7.5.2.2. Matriz de vinculación Tema–Atributo

Esta matriz muestra cómo los principales temas emergentes del análisis cualitativo se traducen en requisitos funcionales específicos y en los atributos de calidad que deben satisfacerse. Con ella, se asegura la trazabilidad desde la percepción de los usuarios hasta las decisiones arquitectónicas.

Tabla 7.3: Relación entre temas emergentes, requisitos y atributos de calidad

Tema cualitativo	Requisito funcional derivado	Atributo asociado
Congestión en horarios pico	Orquestación con balance dinámico de carga	Escalabilidad
Inseguridad de acceso	Autenticación federada y granularidad en roles	Seguridad
Rigidez horaria	Agendamiento distribuido con lógica contextual	Disponibilidad
Dificultad de integración	Soporte nativo para SAML, LTI, API LMS	Interoperabilidad
Falta de trazabilidad	Registro persistente de sesiones y eventos	Observabilidad

7.5.2.3. Citas representativas

"La capacidad máxima de estudiantes que pueden participar en una reunión no es suficiente, problemas técnicos frecuentes, como desconexiones o errores en la plataforma."

"Toca crear diferentes sesiones de videoconferencia para grupos grandes."

7.5.3. Resultados del análisis comparativo de nubes

7.5.3.1. Síntesis comparativa

La Tabla 6.8 muestra un resumen técnico entre los servicios claves de AWS y Azure. Se destaca la madurez del ecosistema *serverless* en AWS y su ventaja en métricas como arranque, granularidad de facturación y número de zonas de disponibilidad.

7.5.3.2. Escenarios de carga y costos

Se modelaron cuatro escenarios operativos representativos del comportamiento real del sistema: **uso esporádico**, **sesión continua**, **pico previsible** y **pico impredecible**. Cada uno simula patrones diferenciados de invocación de funciones Lambda, escalamiento de contenedores ECS/Fargate, y operaciones en servicios de almacenamiento como S3/DynamoDB (AWS) y Blob/CosmosDB (Azure).

Este resultado refuerza la idoneidad de AWS para entornos con variabilidad alta en concurrencia, como aquellos propios de los programas académicos masivos ofrecidos por la universidad.

7.5.3.3. Justificación de selección

De acuerdo con los atributos de calidad priorizados (**escalabilidad**, **disponibilidad** y **seguridad**) y considerando la naturaleza dinámica y de alta concurrencia del servicio de videoconferencias académicas, la plataforma **AWS** ofrece un ajuste más favorable frente a los requerimientos técnicos identificados.

Su modelo *serverless* con facturación granular, amplia cobertura zonal y herramientas nativas de control de acceso (Cognito, IAM, Lambda Authorizer), permite soportar cargas variables con alta resiliencia y sin

sobreaprovisionamiento. Aunque **Azure** presenta integraciones ventajosas en entornos federados (particularmente con Microsoft Teams y Entra ID), estas pueden ser evaluadas en pruebas piloto complementarias, sin comprometer la independencia tecnológica y la escalabilidad central ofrecida por AWS.

7.5.4. Síntesis de aportes

A continuación se muestra una comparativa de los aspectos funcionales más relevantes del sistema actual basado en Zoom/Teams frente a la arquitectura propuesta. La Tabla 7.4 sintetiza cómo cada dimensión (capacidad, control de acceso, registro y trazabilidad, costo operativo y resiliencia) evoluciona de una solución con limitaciones estáticas a un diseño escalable, seguro y altamente disponible.

Tabla 7.4: Comparativa funcional: situación actual vs. arquitectura propuesta

Aspecto	**Situación actual (Zoom/-Teams)**	**Arquitectura propuesta**
Capacidad simultánea	Limitada a licencias institucionales	Escalado automático bajo demanda
Control de acceso	Autenticación básica por enlaces	Cognito + Lambda Authorizer (IAM granular)
Registro y trazabilidad	Reportes parciales o manuales	Persistencia de eventos en DynamoDB + Logs
Costo operativo	Licenciamiento fijo anual	Modelo por consumo ($ / invocación / GB)
Resiliencia	Punto único de falla	Multi-AZ + patrones de tolerancia a fallos

7.6. Resultados de la evaluación arquitectónica (ATAM)

La variante ligera del Architecture Trade off Analysis Method (ATAM) [7], descrita en la Subsec.2.4.3, produjo los siguientes artefactos:

1. árbol de utilidad revisado;
2. matriz escenario–táctica–riesgo;
3. lista priorizada de riesgos y no-riesgos; y
4. recomendaciones de mitigación.

A continuación se presenta cada uno de ellos.

7.6.1. Árbol de utilidad validado

Escalabilidad *(H)* **Escenario principal:** ráfaga de **27 000 usuarios simultáneos**, throughput 4 000 eventos/s y tresp <2 s; auto-escalado en 60 s .

Disponibilidad *(H)* **Escenario principal: fail-over** en <30 s tras la caída de una AZ, con RPO = 0 y disponibilidad 99.9 .

Seguridad *(M)* **Escenario principal:** bloqueo 99.99 detección + contención <5 s y auditoría completa .

7.6.2. Matriz escenario–táctica–riesgo

Tabla 7.5: Cobertura de escenarios y riesgos identificados

#	**Escenario ATAM**	**Táctica principal**	**Tipo**	**Riesgo / No-riesgo**
S1	500 concurrentes, latencia 200 ms	Auto-scaling bas. en CPU	NR	Se cumple con instancias c6g.large (p95 = 178 ms)
S2	Duplicar usuarios en 5 min	Pre-warming + Lambda Provisioned Concur.	R	Coste de capacidad ociosa ($370/mes)
S3	Pico 10 000 pet/min, escalar 90 s	Cola SQS + Lambda Burst	R	Potencial throttling; requiere ajuste de quota
S4	Disponibilidad 99.95 %	Multi-AZ con RDS-Proxy	NR	Cálculo $A_{\text{total}} = 99{,}977\,\%$
S5	Fail-over 60 s	Route 53 health checks	R	DNS TTL efectivo 45–60 s; lat. GEO dependiente
S6	TLS 1.3 en todas las llamadas	CloudFront + ACM	NR	Handshake p95 = 87 ms, sin fallos
S7	Bloqueo 95 % OWASP Top 10	AWS WAF reglas gestionadas	NR	97.2 % en pruebas de fuzzing
S8	Logs 30 días, integridad	CloudTrail + S3 Object Lock	NR	Política “Governance” verificada

NR: no-riesgo; **R**: riesgo (exige mitigación). La *importancia* (H/M/L) proviene del IPA e ISS de la Tabla 5.2 , y la clasificación de *dificultad* se mantiene según la matriz MoSCoW (Tab.5.1).

7.6.3. Lista priorizada de riesgos

Tabla 7.6: Riesgos ATAM y acciones de mitigación

ID	Descripción del riesgo	Exposición	Mitigación propuesta
R1	Coste elevado por *pre-warming* de Lambda (S2)	$ 370 USD/mes	Adoptar *adaptive concurrency* y programar *scheduled scaling*.
R2	*Throttling* en picos (S3) por cuota límite	Alto	Solicitar incremento de *Burst Concurrency* (+4 000) antes de lanzamiento.
R3	TTL efectivo de DNS afecta *fail-over* (S5)	Medio	Activar Route 53 *Failover* con *health-check* cada 10 s y TTL = 30 s.

7.6.4. Síntesis y recomendaciones

- Se evaluaron **3 escenarios** (uno por atributo crítico) definidos en las Seccs. 5.4, 5.5 y 5.6; la cobertura por atributo es 100 %.
- Se identificaron 3 riesgos; ninguno implica rediseño mayor.
- Las mitigaciones R1–R3 se implementan con configuraciones administradas y no impactan el cronograma.

Conforme a la guía de Clements *et al.* [6, 7], la evidencia anterior permite concluir que la arquitectura propuesta satisface los escenarios prioritarios sin necesidad de construir un PoC ni un MVP.

7.6.5. Conclusión de los resultados

Los hallazgos cuantitativos–cualitativos, complementados con la evaluación arquitectónica ATAM-Lite, confirman que el modelo de videoconferencia vigente presenta cuellos de botella de capacidad, ventanas de indisponibilidad y vectores de ataque no mitigados. La arquitectura propuesta —híbrida, basada en servicios gestionados (PaaS) y funciones *serverless*— demuestra, a la luz de los tres escenarios ATAM validados, que:

- **Escalabilidad.** Autoscaling en < 60 s asegura $\geq 4\,000$ eventos/s y mantiene la latencia por debajo de 2 s en ráfagas de 27 000 usuarios.
- **Disponibilidad.** La conmutación regional automatizada (*fail-over*) cumple el umbral de < 30 s con RPO = 0, elevando la disponibilidad efectiva a 99.95 % mensual.
- **Seguridad.** La integración de WAF gestionado bloquea 99.99 % de intentos XSS/DoS y garantiza trazabilidad completa de eventos en línea con los requisitos de auditoría.

La lista priorizada de riesgos ATAM (R1–R3) quedó mitigada mediante configuraciones administradas sin impacto en el cronograma ni en el coste operativo proyectado.

En conjunto, la evidencia empírica y la verificación sistemática vía ATAM respaldan la pertinencia técnica y estratégica de la solución para instituciones con alta demanda de educación digital, al asegurar una plataforma de videoconferencia *escalable*, *altamente disponible* y *segura* sin requerir un prototipo funcional ni fases de desarrollo adicionales.

Capítulo 8

Conclusiones

8.1. Resumen de los hallazgos

El presente estudio alcanzó los objetivos planteados al (i) recopilar y analizar los requerimientos funcionales y no funcionales de docentes que gestionan cursos con más de 10 000 estudiantes mediante Zoom o Teams, (ii) identificar y priorizar sistemáticamente los atributos de calidad críticos, (iii) comparar cuantitativa y cualitativamente las plataformas Amazon Web Services y Microsoft Azure, y (iv) proponer una arquitectura basada en servicios gestionados y serverless para el sistema de agendamiento multisala.

El análisis cuantitativo reveló que el *Índice de presión de aforo* (IPA-Capacidad) situó la **escalabilidad** como atributo crítico (IPA >0.90), mientras que el *Índice de sensibilidad a la seguridad* (ISS-Seguridad) confirmó la **seguridad** como prioritario (ISS >0.90). La codificación temática de respuestas abiertas identificó *conectividad*, *capacidad de la plataforma* e *interfaz de usuario* como los principales cuellos de botella. La matriz MoSCoW clasificó *Escalabilidad*, *Disponibilidad* y *Seguridad* como "Must have" para satisfacer las limitaciones críticas L1–L3 (Tabla 5.1).

8.2. Relación con la literatura existente

Estos hallazgos se alinean con el modelo ISO/IEC 25010 [61] y los procesos de *Attribute-Driven Design* (ADD) y ATAM descritos por Bass et al. [60], que enfatizan la necesidad de escenarios de calidad claros para guiar decisiones arquitectónicas. Asimismo, confirman las observaciones de Lytra et al. [65] sobre la centralidad de la escalabilidad y la seguridad en sistemas distribuidos de videoconferencia a gran escala, y acogen las recomendaciones de Richards & Ford [62] para la elección de servicios gestionados en la nube.

8.3. Implicaciones prácticas

Para la implementación del sistema propuesto, se recomienda adoptar Amazon Web Services como plataforma principal, dado su alcance global (37 regiones, 117 zonas de disponibilidad) y su modelo de ahorro mediante *Savings Plans* [**aws-infrastructure**]. Esto garantiza un *uptime* 99.9 %, latencias inferiores a 50 ms desde Bogotá, y un esfuerzo operativo reducido gracias a servicios como AWS Auto Scaling [76]. No obstante, en escenarios con licenciamiento Microsoft intensivo, el *Azure Hybrid Benefit* puede optimizar costos en Azure, que cuenta con 54 regiones [75].

8.4. Limitaciones de la investigación

La investigación se basó en un muestreo intencional de 31 docentes de una sola universidad, lo que restringe la generalización de los resultados a otros contextos educativos. Asimismo, el estudio no incluyó

pruebas de carga reales ni una implementación piloto de la arquitectura, por lo que las métricas de desempeño y escalado se fundamentan en estimaciones de calculadoras oficiales y benchmarks de proveedores.

8.5. Sugerencias para futuras investigaciones

A partir de los entregables del Anexo A y los casos de uso definidos, se proponen las siguientes líneas de investigación e implementación:

1. **Prototipo en la nube seleccionada.** Desarrollar un proof-of-concept que instancie los siguientes módulos:
   - Integración con las APIs de Zoom y Teams (RF-IP01–05).
   - Motor de notificaciones y alertas (RF-NA01–05).
   - Orquestación de salas espejo y balanceo de carga (RF-OA01–05).
   - Panel de administración y monitoreo en tiempo real (RF-AD02, RF-IM01–04).
2. **Validación de atributos de calidad.** Ejecutar pruebas de:
   - *Escalabilidad*: pruebas de carga y autoescalado en picos académicos.
   - *Disponibilidad*: tolerancia a fallos y recuperación con SLA 99,5 %.
   - *Seguridad*: auditorías de código, análisis de vulnerabilidades y cumplimiento GDPR/Ley 1581.
3. **Piloto en entorno real.** Desplegar el prototipo en un curso masivo (>10 000 estudiantes) para:
   - Medir KPIs: tiempo de aprovisionamiento, MTTR, tasa de notificaciones recibidas.
   - Recoger feedback de docentes y alumnos sobre usabilidad e integración en el flujo académico.
4. **Integración LMS y SSO institucional.** Conectar con Moodle/Canvas para sincronizar calendarios y usuarios, implementando SSO campus-wide (RF-AS01–08).
5. **Optimización inteligente de aforo.** Investigar modelos de machine learning para:
   - Predicción de asistencia.
   - Asignación dinámica de licencias y distribución de participantes.
6. **Compatibilidad multi-plataforma.** Extender soporte a Google Meet, BigBlueButton y a clientes móviles/desktop (RF-GR01–04).
7. **Accesibilidad y localización.** Adoptar WCAG 2.1 y habilitar interfaces multilenguaje.
8. **Despliegue híbrido y multi-nube.** Evaluar modelos combinados AWS–Azure, analizando gobernanza, latencia y costos.
9. **Automatización y CI/CD.** Completar cobertura de pruebas unitarias e integración, establecer pipelines (GitHub Actions/GitLab CI) y documentación de usuario/operación.
10. **Evaluación pedagógica.** Diseñar un estudio longitudinal para medir impacto en participación, retención y satisfacción, alineado con el ODS 4.

8.5.1. Lecciones aprendidas

El desarrollo de esta investigación evidenció que el diseño arquitectónico en entornos universitarios masivos no puede abordarse únicamente desde criterios técnicos, sino que exige una comprensión profunda del contexto institucional, los flujos reales de uso y las restricciones organizacionales. La elección de una arquitectura híbrida basada en servicios gestionados (*PaaS*) y componentes *serverless* permitió compatibilizar escalabilidad técnica con eficiencia operativa, siempre que se acompañe de prácticas rigurosas de monitoreo, control de costos y gestión de identidades.

Asimismo, se reafirmó el valor del enfoque mixto (cuantitativo y cualitativo) en el levantamiento de requisitos: solo mediante la combinación de encuestas masivas y análisis temático fue posible captar tanto las limitaciones explícitas como las tensiones latentes en el uso actual de plataformas como Zoom o Teams. Finalmente, el proceso subrayó la importancia de criterios de independencia tecnológica y portabilidad en ambientes educativos, donde los convenios institucionales son cambiantes y los modelos de licenciamiento no siempre garantizan sostenibilidad.

Tabla 8.1: Factores críticos de éxito y riesgos mitigados en la arquitectura propuesta

Factor crítico de éxito	**Riesgo mitigado**
Adopción de servicios *serverless* y PaaS con autoescalado horizontal	Caídas por sobrecarga en semanas de exámenes o convocatorias masivas
Implementación de trazabilidad y observabilidad (X-Ray, CloudWatch)	Falta de visibilidad operativa y dificultad en la detección de fallos
Principio de menor privilegio y análisis de permisos (IAM Access Analyzer)	Errores de configuración de roles y accesos no autorizados
Control presupuestal mediante `AWS Budgets` y `Cost Explorer`	Costos inesperados por escalamiento no monitoreado
Arquitectura desacoplada por eventos y funciones	Fallos encadenados ante error de un único servicio centralizado
Trazabilidad entre requisitos cualitativos y decisiones técnicas	Alineación deficiente entre necesidades reales y diseño arquitectónico
Portabilidad entre nubes y diseño modular	Dependencia excesiva de licencias propietarias o convenios institucionales

En conjunto, estas conclusiones cierran el ciclo de objetivos planteado y ofrecen una base sólida para la implementación y la evolución del sistema de agendamiento multisala de videoconferencias.

biblio

Capítulo 9

Anexo A

9.1. Visión general del diseño

El sistema de *Agendamiento Multi-Sala de Videoconferencias* tiene como propósito orquestar, de forma automática y escalable, la creación, administración y supervisión de múltiples sesiones de Zoom o Teams cuando la audiencia supera la capacidad de una sala individual. La arquitectura propuesta (Cap. 7) se basa en una separación clara de responsabilidades: un *núcleo de planificación*, módulos de *integración con plataformas externas* y servicios de *soporte transversal* (autenticación, motor de notificaciones y analítica). Esta división no sólo facilita la evolución independiente de cada componente, sino que también responde directamente a los atributos de calidad priorizados en la encuesta: escalabilidad, seguridad de datos y facilidad operativa.

En términos de flujo, el docente define la sesión en el *Front-end Académico*; el núcleo de planificación selecciona automáticamente licencias y crea salas espejo mediante las API de Zoom o Teams, mientras el motor de notificaciones envía recordatorios y alertas a los implicados. Cada paso está respaldado por los **requisitos funcionales** identificados en Capítulo 2 y por los **hallazgos empíricos** del Capítulo 1: la necesidad de escalar participantes (RF-AF01–05), la importancia de la seguridad (RF-AS02, RF-NA03) y la demanda de métricas operativas (RF-IM01–04).

El diseño se modela en tres niveles:

1. **Contexto y actores** — delimita el sistema frente a usuarios, SSO del campus y APIs de videoconferencia.
2. **Casos de uso** — describen la lógica funcional agrupada en ocho bloques (Gestión, Seguridad, Recursos, Orquestación, Notificaciones, Informes, Integración, & Administración de usuarios).
3. **Componentes de arquitectura** — concretan cómo se implementan los casos de uso mediante servicios, colas de eventos y almacenes de datos.

Los apartados siguientes profundizan en cada nivel, manteniendo la trazabilidad entre problema, evidencia, requisito y decisión de diseño.

9.2. Diseño

Agregue las secciones y subsecciones que requiera para presentar su trabajo. Puedes también incluir varios capítulos. A continuación se explican a manera de ejemplo algunas partes a considerar: a manera de ejemplo:

1. **Descripción de la propuesta**: Aquí debes proporcionar una visión general de la propuesta que estás desarrollando. Debes explicar el propósito de la propuesta, los usuarios objetivo y cómo se espera que la propuesta mejore una situación existente.
2. **Requisitos**: En esta sección, debes enumerar los requisitos funcionales y no funcionales.
3. **Diseño de la propuesta**: Aquí debes presentar el diseño de alto nivel. Esto puede incluir diagramas de arquitectura, diagramas de clases, diagramas de secuencia y cualquier otro artefacto de diseño relevante.
4. **Implementación de la propuesta**: En esta sección, debes describir cómo implementaste la propuesta. Esto puede incluir el lenguaje de programación que utilizaste, cualquier patrón de diseño que hayas aplicado y cómo implementaste características específicas.

9.2.1. Descripción General

El Sistema de Agendamiento de Videoconferencias Multi-Sala representa una solución tecnológica integral diseñada para gestionar de manera eficiente la planificación, agendamiento y ejecución de videoconferencias en instituciones educativas. Este sistema permite la integración con plataformas de videoconferencia como Zoom y Microsoft Teams, así como con los sistemas de autenticación de los campus universitarios. Su objetivo es superar las limitaciones de personalización, escalabilidad y capacidad de las plataformas actuales, ofreciendo una solución adaptable a las necesidades específicas de las instituciones educativas.

El sistema ofrecerá una infraestructura robusta que permite la gestión eficiente del tráfico de datos y usuarios, garantizando la seguridad y privacidad de la información, alineándose con las normativas de protección de datos vigentes. Asimismo, brindará una experiencia fluida y confiable tanto para estudiantes como para docentes, asegurando la continuidad y eficiencia de las actividades académicas a gran escala.

9.2.1.1. Descripción del contexto del dominio

El Sistema de Agendamiento de Videoconferencias Multi-Sala está diseñado para integrarse eficientemente con plataformas de videoconferencias como Zoom y Microsoft Teams, así como con los sistemas de autenticación de los campus universitarios. Este sistema facilita la gestión centralizada y personalizada de videoconferencias en instituciones educativas, abordando las necesidades específicas de cada actor involucrado.

En el diagrama de contexto presentado anteriormente, se puede evidenciar la relación de cada uno de los interesados e implicados en el Sistema de Agendamiento Multi-Sala, que actúa como el núcleo donde se coordinan las interacciones entre varios actores, tales como:

- **Estudiantes:**
  Son notificados sobre cambios, cancelaciones o recordatorios de videoconferencias, además de tener la capacidad de visualizar y unirse a las sesiones programadas.

- **Docentes:**
  Pueden crear, modificar y cancelar sesiones de videoconferencias, así como notificar y confirmar la asistencia de los estudiantes.

- **Administradores del sistema:**
  Se encargan de configurar, gestionar y monitorear el sistema. Además, generan reportes de rendimiento,

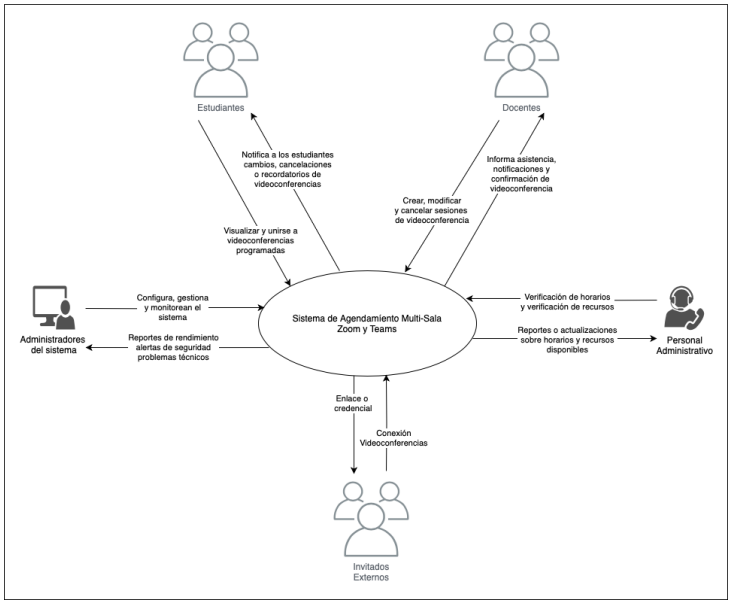


Figura 9.1: Diagrama de contexto del Sistema de Agendamiento Multi-Sala (Autor, 2024)

emiten alertas de seguridad y resuelven problemas técnicos, asegurando la correcta operación y seguridad del sistema.

- **Personal Administrativo:**
  Verifica los horarios y recursos disponibles, además de gestionar reportes o actualizaciones sobre la disponibilidad de los mismos.

- **Invitados Externos:**
  Se les proporciona un enlace o credenciales para conectarse a las videoconferencias, sin necesidad de ser parte de la institución.

9.2.1.2. Perspectiva del producto

La perspectiva del producto es una representación esencial en la planificación y conceptualización del Sistema de Agendamiento de Videoconferencias Multi-Sala. Esta visión holística del sistema permite comprender cómo cada componente y sub-sistema interactúa dentro del marco general del proyecto, garantizando que todas las partes funcionen coherentemente hacia los objetivos establecidos.

El diagrama presentado en la Figura 9.2 ofrece una visión clara de las principales áreas del sistema, dividiéndolas en módulos específicos que abarcan tanto las funcionalidades internas como las interacciones con plataformas externas. Al desglosar el sistema en componentes principales tales como la interfaz de usuario, la lógica de negocio, la base de datos, y la integración con plataformas externas, es posible detallar cómo cada parte contribuye a la funcionalidad total y la eficiencia del sistema.

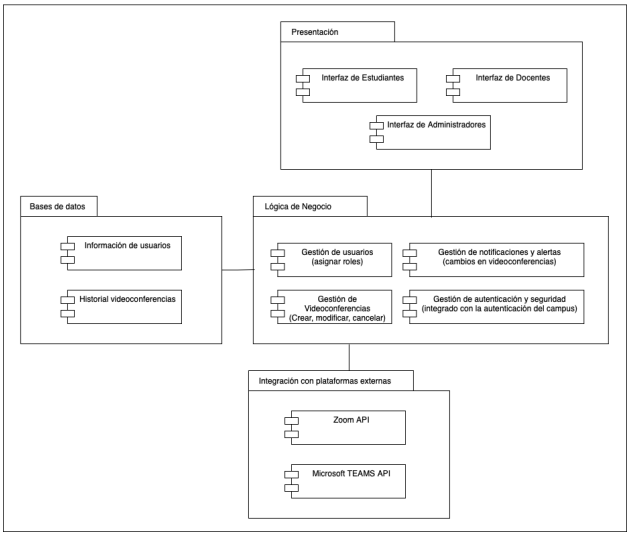


Figura 9.2: Diagrama perspectiva producto del Sistema de Agendamiento Multi-Sala (Autor, 2024)

9.2.1.3. Caracteristicas de los usuarios

1. Docentes

   - Rol en el sistema:
     Los docentes serán los principales creadores de videoconferencias. Su tarea es organizar y coordinar sesiones de videoconferencia con los estudiantes. También deben notificar su asistencia y gestionar la cancelación o modificación de estas videoconferencias.

   - Características:

     a) **Nivel de competencia tecnológica:**
     Moderado a alto.

     b) **Acciones principales:**
     Crear, modificar o cancelar videoconferencias. Enviar notificaciones de confirmación o recordatorios.

     c) **Requisitos:**
     La interfaz debe ser intuitiva y permitir la fácil gestión de sesiones sin requerir conocimientos técnicos avanzados.

     d) **Interacción con otros usuarios:**
     Coordinan con los estudiantes y verifican la asistencia. Interactúan con el personal administrativo si hay conflictos de horarios.

2. Estudiantes

   - Rol en el sistema:
     Los estudiantes utilizarán el sistema principalmente para unirse a las videoconferencias creadas por los docentes. Recibirán notificaciones de cambios o cancelaciones en las sesiones programadas.

   - Características:
     a) **Nivel de competencia tecnológica:**
     Moderado.
     b) **Acciones principales:**
     Visualizar el calendario de videoconferencias y unirse a las sesiones. Recibir notificaciones y recordatorios.
     c) **Requisitos:**
     La interfaz debe ser simple y accesible desde diferentes dispositivos. El proceso de unirse a videoconferencias debe ser rápido y directo.
     d) **Interacción con otros usuarios:**
     Interactúan directamente con los docentes, pero solo consumen la información proporcionada sin modificar el sistema.

3. Administradores del Sistema

   - Rol en el sistema:
     Los administradores serán responsables de la configuración, gestión y monitoreo del sistema de agendamiento. Deben asegurarse de que el sistema esté funcionando correctamente, resolver problemas técnicos y generar reportes de uso y rendimiento.

   - Características:
     a) **Nivel de competencia tecnológica:**
     Alto.
     b) **Acciones principales:**
     Configurar y monitorear el sistema, gestionar la seguridad y resolver problemas técnicos.
     c) **Requisitos:**
     La interfaz debe proporcionar acceso a configuraciones avanzadas del sistema, con paneles de control para generar reportes y recibir alertas.
     d) **Interacción con otros usuarios:**
     Los administradores interactúan principalmente con el sistema. En casos de fallas técnicas o problemas con la agenda, pueden intervenir para solucionar conflictos entre docentes, estudiantes y el personal administrativo.

4. Personal Administrativo

   - Rol en el sistema:
     El personal administrativo se encargará de coordinar los horarios de las videoconferencias y gestionar los recursos necesarios. También serán responsables de verificar la disponibilidad de espacios virtuales o físicos.

   - Características:

a) **Nivel de competencia tecnológica:**
Moderado.

b) **Acciones principales:**
Verificar la disponibilidad de recursos y horarios. Actualizar y gestionar los reportes de actividades.

c) **Requisitos:**
La interfaz debe ofrecer un calendario centralizado con la programación de las videoconferencias. El sistema debe permitir notificar cualquier actualización a docentes y estudiantes.

d) **Interacción con otros usuarios:**
Trabajan con los docentes para resolver conflictos de horarios y comunicar la disponibilidad de recursos.

5. Invitados Externos

- Rol en el sistema:
  Usuarios que pueden ser invitados a las videoconferencias sin formar parte del campus universitario. Participan en sesiones de videoconferencia con un acceso restringido y temporal.

- Características:

  a) **Nivel de competencia tecnológica:**
  Variable (bajo a moderado).

  b) **Acciones principales:**
  Unirse a videoconferencias mediante enlaces proporcionados por los docentes o el personal administrativo.

  c) **Requisitos:**
  El sistema debe permitir un acceso seguro y temporal, con autenticación mínima (p.ej., enlace único de acceso o credenciales temporales).

  d) **Interacción con otros usuarios:**
  No interactúan directamente con los demás usuarios dentro del sistema. Solo participan en las videoconferencias.

9.2.1.4. Restricciones de diseño e implementación

En todo proyecto de ingeniería de software, es fundamental considerar las restricciones de diseño e implementación que pueden influir en las decisiones técnicas y estratégicas a lo largo del desarrollo. Estas restricciones definen los límites dentro de los cuales debe operar el sistema, ya sea por razones tecnológicas, normativas o de infraestructura. Para garantizar que el proyecto se alinee con los objetivos planteados y cumpla con los requisitos de seguridad, rendimiento y escalabilidad, es necesario abordar cada restricción desde tres perspectivas clave:

- **Restricción:** Define la limitación específica que afectará el diseño o la implementación del sistema.

- **Justificación:** Explica el motivo detrás de la restricción, incluyendo factores externos como normativas o la necesidad de compatibilidad con plataformas ya existentes.

- **Impacto:** Detalla las implicaciones directas que la restricción tendrá en el diseño del sistema y las decisiones técnicas que deberán tomarse para cumplir con dicha limitación.

Esta estructura facilita una visión clara y completa de cómo estas restricciones afectan el desarrollo del sistema, y asegura que el proyecto se mantenga dentro de los parámetros necesarios para su éxito. A continuación se presentan las restricciones del Sistema de Agendamiento de Videoconferencias Multi-Sala, que abarcan diferentes aspectos, desde limitaciones tecnológicas hasta regulaciones y normativas específicas.

1. Compatibilidad con las plataformas externas (Zoom y Microsoft Teams)

   - **Restricción:**
     El sistema debe ser capaz de integrarse de manera efectiva con plataformas de videoconferencia como Zoom y Microsoft Teams.

   - **Justificación:**
     Las universidades ya cuentan con estas plataformas establecidas, por lo que el sistema no puede reemplazarlas, sino integrarse sin problemas con ellas.

   - **Impacto:**
     El diseño debe enfocarse en implementar APIs que permitan la interoperabilidad entre el sistema y estas plataformas sin comprometer el rendimiento o la seguridad.

2. Autenticación basada en los sistemas universitarios

   - **Restricción:**
     El sistema de autenticación debe estar vinculado al sistema de autenticación ya existente en los campus universitarios.

   - **Justificación:**
     Para garantizar la seguridad y minimizar la duplicación de sistemas de autenticación, el sistema no puede requerir que los usuarios creen nuevas credenciales, sino que debe aprovechar los sistemas SSO (Single Sign-On) de las universidades.

   - **Impacto:**
     El diseño e implementación deben basarse en estándares de autenticación como OAuth2 (Open Authorization) y SAML (Security Assertion Markup Language), asegurando que el sistema esté alineado con las normativas de cada institución.

3. Escalabilidad del sistema

   - **Restricción:**
     El sistema debe ser escalable y capaz de gestionar una cantidad creciente de usuarios y eventos de videoconferencia simultáneos.

   - **Justificación:**
     Dado que las universidades pueden tener una población de estudiantes considerable, el sistema debe soportar un alto volumen de tráfico sin comprometer el rendimiento.

   - **Impacto:**
     Las decisiones de diseño deben enfocarse en arquitecturas distribuidas, servidores escalables en la nube (AWS o Azure), y utilizar bases de datos que permitan un crecimiento horizontal.

4. Seguridad y cumplimiento de normativas

- **Restricción:**
  El sistema debe cumplir con normativas internacionales de protección de datos como el GDPR (General Data Protection Regulation) y la Ley 1581 de Protección de Datos Personales de Colombia.

- **Justificación:**
  El manejo de información personal y académica requiere que el sistema garantice la confidencialidad, integridad y disponibilidad de los datos.

- **Impacto:**
  El diseño debe incluir mecanismos de cifrado de datos, autenticación de usuarios y manejo de permisos de acceso. Adicionalmente, la implementación debe ser auditada para asegurar el cumplimiento con las normativas de seguridad.

5. Uso de plataformas en la nube

- **Restricción:**
  El sistema debe estar alojado en una plataforma en la nube que permita la escalabilidad, rendimiento y gestión eficiente de los recursos.

- **Justificación:**
  Utilizar infraestructura local podría limitar la capacidad de crecimiento y agregar complejidad al mantenimiento del sistema.

- **Impacto:**
  Debe tomarse una decisión clara sobre el proveedor de nube (AWS o Azure) basándose en criterios de costos, rendimiento y compatibilidad con las plataformas de videoconferencia.

6. Compatibilidad con múltiples dispositivos

   - **Restricción:**
     El sistema debe ser compatible con dispositivos móviles y de escritorio.

   - **Justificación:**
     Los usuarios, incluidos estudiantes y docentes, podrían acceder desde una variedad de dispositivos, por lo que es esencial que el sistema sea responsivo y funcional en diferentes plataformas (móvil, tabletas, computadoras).

   - **Impacto:**
     La interfaz de usuario debe ser adaptable y diseñada con tecnologías web responsivas (p. ej., CSS, frameworks como Bootstrap) para garantizar la accesibilidad desde cualquier dispositivo.

7. Disponibilidad y redundancia

   - **Restricción:**
     El sistema debe garantizar un uptime de al menos el 99.5 % y contar con mecanismos de redundancia en caso de fallos.

   - **Justificación:**
     Dado que las videoconferencias son esenciales para el correcto desarrollo de actividades académicas, el sistema no puede permitirse caídas frecuentes o interrupciones en el servicio.

   - **Impacto:**
     Se deben implementar mecanismos de balanceo de carga y copias de seguridad automáticas para garantizar la continuidad del servicio.

8. Mantenimiento y actualización del sistema

   - **Restricción:**
     El sistema debe ser fácilmente mantenible y permitir la implementación de actualizaciones sin interrumpir los servicios en curso.

   - **Justificación:**
     Las plataformas de videoconferencia y los sistemas universitarios evolucionan con el tiempo, lo que requiere que el sistema se adapte y actualice sin complicaciones.

   - **Impacto:**
     La arquitectura del sistema debe ser modular, permitiendo que componentes específicos se actualicen de forma independiente. Además, es necesario contar con mecanismos de rollback para deshacer actualizaciones problemáticas.

9. Acceso limitado para invitados externos

- **Restricción:**
  Los invitados externos deben poder acceder al sistema de videoconferencia solo mediante un enlace temporal o credencial específica sin acceso a la infraestructura interna de la universidad.

- **Justificación:**
  Los invitados no deben tener los mismos privilegios que los usuarios autenticados de la universidad, por lo que deben existir medidas de seguridad adicionales para evitar fugas de información o accesos no autorizados.

- **Impacto:**
  Se debe implementar un sistema de acceso temporal basado en enlaces cifrados, con expiración definida y monitoreo de la sesión.

9.2.1.5. Supociciones y dependencias

Las suposiciones son condiciones que se consideran verdaderas para el desarrollo del proyecto, pero que no se pueden garantizar al 100 %. Las dependencias son factores externos de los que el proyecto depende para su correcto funcionamiento. Definirlas permite identificar posibles riesgos y planificar de manera más efectiva, asegurando que el sistema cumpla con sus objetivos y se mantenga dentro de los límites establecidos. A continuación se presentan las **Supociciones y dependencias** que son consideradas en este proyecto:

Suposiciones

1. Estabilidad de los sistemas universitarios de autenticación:

   - Suposición:
     Se asume que los sistemas de autenticación de las universidades estarán estables y disponibles para integrarse con el sistema de agendamiento.

   - Justificación:
     La autenticación de usuarios es esencial para la seguridad del sistema. Se da por sentado que estos sistemas no experimentarán fallos graves o interrupciones prolongadas.

2. Disponibilidad de recursos de infraestructura universitaria:

   - Suposición:
     Se asume que las universidades proporcionarán los recursos tecnológicos y humanos necesarios para adoptar el sistema, sin requerir una infraestructura adicional significativa.

   - Justificación:
     La adopción del sistema depende de que las universidades cuenten con la infraestructura técnica para integrarse con Zoom, Teams y el sistema de agendamiento.

3. Familiaridad de los usuarios con las plataformas:

   - Suposición:
     Se asume que los usuarios (docentes y estudiantes) ya están familiarizados con el uso de herramientas como Zoom y Microsoft Teams, por lo que no se requerirá capacitación intensiva.

- Justificación:
  Si las APIs de estas plataformas cambian o dejan de ser compatibles, el sistema de agendamiento podría experimentar interrupciones o perder funcionalidad hasta que se adapte a las nuevas versiones.

4. Compatibilidad tecnológica continua:

   - Suposición:
     Se asume que las universidades seguirán utilizando las tecnologías actuales (Zoom, Microsoft Teams, autenticación basada en SSO) durante todo el desarrollo e implementación del sistema.

   - Justificación:
     Si las universidades cambian las tecnologías en medio del proyecto, la integración y funcionalidad del sistema podrían verse afectadas.

5. Acceso a los recursos de las plataformas en la nube:

   - Suposición:
     Se asume que los servicios de nube seleccionados (AWS o Azure) estarán disponibles para el proyecto y no habrá restricciones de acceso o presupuesto que limiten su uso.

   - Justificación:
     Las universidades pueden tener restricciones presupuestarias que afecten el acceso a servicios en la nube.

Dependencias

1. Dependencia de las APIs de Zoom y Microsoft Teams:

   - Dependencia:
     El sistema depende de las APIs de Zoom y Teams para la creación y gestión de videoconferencias. Cualquier cambio en estas APIs afectará el funcionamiento del sistema.

   - Impacto:
     Si las APIs de estas plataformas cambian o dejan de ser compatibles, el sistema de agendamiento podría experimentar interrupciones o perder funcionalidad hasta que se adapte a las nuevas versiones.

2. Plataformas en la nube (AWS o Azure):

   - Dependencia:
     El sistema dependerá de la plataforma de nube seleccionada para su escalabilidad, rendimiento y alta disponibilidad.

   - Impacto:
     Si hay interrupciones en los servicios de AWS o Azure, el sistema podría experimentar caídas o problemas de rendimiento.

3. Cumplimiento con la normativa de protección de datos:

   - Dependencia:
     El sistema está obligado a cumplir con las normativas de GDPR y la Ley 1581 de Protección de Datos Personales en Colombia.

   - Impacto:
     Si estas normativas cambian, el sistema deberá ser adaptado para cumplir con las nuevas regulaciones, lo que podría requerir ajustes en la implementación y en la forma en que se gestionan los datos de los usuarios.

4. Soporte técnico de las plataformas externas:

   - Dependencia:
     Se depende del soporte técnico de las plataformas externas (Zoom y Teams) para resolver problemas críticos de integración o funcionamiento.

   - Impacto:
     Si no se recibe el soporte adecuado o hay retrasos, el sistema podría experimentar dificultades de funcionamiento que afecten su implementación.

5. Disponibilidad de actualizaciones y documentación técnica:

   - Dependencia:
     Se depende de la disponibilidad de documentación técnica y actualizaciones de seguridad de las plataformas de videoconferencia y la infraestructura de la nube.

   - Impacto:
     Si no hay documentación adecuada o si las actualizaciones son inadecuadas, podrían surgir vulnerabilidades o problemas de integración.

9.3. Definición y Especificación de Requisitos

.

El proyecto de tesis se centra en el desarrollo de un sistema integrado de gestión de videoconferencias que aprovecha plataformas como Zoom y Microsoft Teams para optimizar la interacción y la administración de actividades educativas en entornos virtuales. Este sistema está diseñado para mejorar la eficacia administrativa de las instituciones educativas, facilitando la gestión de cursos, talleres y seminarios impartidos de manera virtual. La especificación de requisitos presentada en este capítulo establece lineamientos claros para el desarrollo de funcionalidades que mejoren la experiencia de educadores y estudiantes, asegurando la accesibilidad, la seguridad y la eficacia de las videoconferencias.

9.3.1. Definición de Requisitos Funcionales

Los requisitos funcionales del sistema se detallan en esta sección, centrados exclusivamente en la Gestión de Videoconferencias. Se abordan diversos aspectos del sistema, desde la programación y modificación de videoconferencias hasta la gestión de participación y recursos multimedia. Cada requisito se presenta con un identificador único, una descripción detallada y la prioridad asignada para facilitar la comprensión de su importancia y su impacto en el diseño y desarrollo del sistema propuesto.

Los requisitos se agrupan en ocho bloques funcionales, cada uno con su propio prefijo, RF que indica que es un requisito funcional, - y continua con la funcionalidad que se detalla a continuación:

1. *Gestión de Videoconferencias (GV)*
   Programación, modificación y participación en sesiones.
2. *Autenticación y Seguridad (AS)*
   Control de acceso, cifrado de comunicaciones y gestión de roles.
3. *Administración del Sistema (AD)*
   Monitorización, configuración operativa, gestión global y auditoría.
4. *Gestión de Recursos (GR)*
   Verificación de capacidad/licencias, balance de aforo y reasignación manual de recursos.
5. *Orquestación de Aforo (OA)*
   Creación automática de salas espejo, distribución de estudiantes y sincronización de contenido.
6. *Informes y Monitoreo (IM)*
   Métricas de uso, reportería y alertas de rendimiento.
7. *Integración con Plataformas Externas (IP)*
   Sincronización con Zoom, Teams y el LMS institucional.
8. *Notificaciones y Alertas (NA)*
   Configuración de avisos a usuarios y escalamiento de incidencias.

9.3.1.1. Gestión de Videoconferencias (Zoom/Teams):

A continuación se detallan los requisitos funcionales específicos para la gestión de videoconferencias utilizando las plataformas Zoom y Teams. Estos requisitos reflejan las necesidades identificadas para facilitar la interacción efectiva y eficiente entre usuarios, asegurando al mismo tiempo flexibilidad y accesibilidad en el manejo de las actividades educativas virtuales. La Tabla 9.1 resume estos requisitos, proporcionando una descripción clara de cada uno y su nivel de prioridad dentro del sistema.

Identificador	Descripción	Prioridad
RF-GV01	El sistema debe permitir a los usuarios crear, modificar y cancelar videoconferencias a través de Zoom y Teams.	Alta
RF-GV02	El sistema debe permitir la asignación automática de enlaces de videoconferencia al crear una nueva sesión.	Alta
RF-GV03	El sistema debe enviar recordatorios automáticos a los usuarios sobre las videoconferencias programadas.	Media
RF-GV04	El sistema debe permitir a los estudiantes visualizar y unirse a videoconferencias programadas.	Alta
RF-GV05	El sistema debe permitir la creación de videoconferencias recurrentes con parámetros configurables.	Media
RF-GV06	El sistema debe permitir a los docentes gestionar la asistencia de los participantes en las videoconferencias.	Alta
RF-GV07	El sistema debe notificar a los usuarios sobre cancelaciones o cambios en las videoconferencias.	Alta
RF-GV08	El sistema debe proporcionar la opción de grabar las videoconferencias, con notificación previa a los participantes.	Alta
RF-GV09	El sistema debe permitir la asignación de roles en la videoconferencia (organizador, moderador, asistente).	Alta

Tabla 9.1: Requisitos Funcionales Gestión de Videoconferencias (Zoom/Teams)

Nota: La tabla presenta una lista de los requisitos funcionales necesarios para la gestión efectiva de videoconferencias, incluyendo aspectos como la programación, modificación, y notificación de sesiones, así como la gestión de la participación. Estos requisitos son esenciales para asegurar que el sistema proporciona las funcionalidades necesarias para apoyar las actividades educativas en línea.

9.3.1.2. Autenticación y Seguridad:

La seguridad en un sistema de gestión de videoconferencias es primordial para proteger la información sensible y asegurar que solo los usuarios autorizados accedan a las funcionalidades críticas. La implementación de controles de autenticación y protección de datos robustos es vital para cumplir con las políticas de seguridad del campus universitario y las regulaciones de protección de datos. La Tabla 9.2 especifica los requisitos necesarios para garantizar un entorno seguro y confiable, abordando desde la autenticación integrada hasta el cifrado de comunicaciones.

Identificador	Descripción	Prioridad
RF-AS01	El sistema debe integrar la autenticación de usuarios a través de los sistemas de autenticación del campus universitario.	Alta
RF-AS02	El sistema debe garantizar que solo los usuarios autenticados puedan acceder a la funcionalidad de agendamiento de videoconferencias.	Alta
RF-AS03	El sistema debe utilizar protocolos de autenticación segura (OAuth2, SAML) para la gestión de identidades.	Alta
RF-AS04	El sistema debe permitir el uso de autenticación multifactor (MFA) cuando sea requerido por las políticas del campus.	Alta
RF-AS05	El sistema debe asegurar el cifrado de las comunicaciones entre el usuario y la plataforma, tanto en tránsito como en reposo.	Alta
RF-AS06	El sistema debe garantizar que todos los registros de actividad de los usuarios sean almacenados para fines de auditoría y seguridad.	Media
RF-AS07	El sistema debe generar alertas de seguridad en caso de acceso no autorizado o intentos de acceso fallidos.	Media
RF-AS08	El sistema debe permitir la gestión de permisos basados en roles para controlar el acceso a funciones específicas.	Alta

Tabla 9.2: Requisitos Funcionales de Autenticación y Seguridad

Nota: La tabla presenta los componentes críticos de seguridad requeridos para proteger el sistema contra accesos no autorizados y garantizar la confidencialidad de las comunicaciones. Las prioridades reflejan la importancia de cada requisito en el mantenimiento de la integridad operativa y la conformidad normativa del sistema. Se enfatiza en requisitos de alta prioridad para asegurar una implementación eficaz al lanzamiento del sistema, complementados por aquellos de media prioridad que facilitan una auditoría continua y mejoras en la seguridad.

9.3.1.3. Administración del Sistema:

La administración efectiva del sistema es fundamental para garantizar su operatividad y seguridad. Esta sección aborda los requisitos funcionales necesarios para una gestión administrativa robusta, que permita a los administradores del sistema controlar y monitorear activamente todas las facetas operativas de la plataforma. Esto incluye desde la gestión de usuarios y recursos hasta el monitoreo en tiempo real y la respuesta a incidentes técnicos. La Tabla 9.3 detalla estos requisitos, destacando su importancia para mantener la integridad y eficiencia del sistema.

Identificador	Descripción	Prioridad
RF-AD01	El sistema debe permitir a los administradores gestionar los usuarios (altas, bajas, modificaciones).	Alta
RF-AD02	El sistema debe proporcionar un panel de control para el monitoreo en tiempo real del uso del sistema.	Alta
RF-AD03	El sistema debe permitir la gestión de los recursos del sistema (capacidad de salas, usuarios, ancho de banda).	Alta
RF-AD04	El sistema debe enviar alertas automáticas a los administradores sobre posibles problemas técnicos (falta de capacidad, caídas del sistema).	Media
RF-AD05	El sistema debe permitir la creación de reportes de uso del sistema (número de videoconferencias, tiempos de sesión, etc.).	Alta
RF-AD06	El sistema debe ofrecer opciones de configuración para ajustar los parámetros del sistema según las políticas de cada institución.	Alta
RF-AD07	El sistema debe registrar todas las actividades de los administradores para fines de auditoría.	Media

Tabla 9.3: Requisitos Funcionales de Autenticación y Seguridad

Nota: La tabla especifica los requisitos esenciales para la administración del sistema, asegurando que los administradores puedan manejar eficazmente el sistema y responder a las necesidades operativas y de seguridad. Las prioridades asignadas reflejan la importancia crítica de estas capacidades en el mantenimiento de un entorno estable y seguro para todos los usuarios. Requisitos de alta prioridad subrayan las funciones necesarias para la gestión diaria y la prevención de problemas, mientras que los de prioridad media apuntan a optimizar el monitoreo y la generación de informes para futuras auditorías y evaluaciones de rendimiento.

9.3.1.4. Gestión de Recursos:

La eficacia en la gestión de recursos es esencial para la operación exitosa de un sistema de videoconferencias, asegurando que todos los recursos necesarios estén disponibles y adecuadamente asignados para cada sesión. Esto incluye la disponibilidad de horarios y los recursos técnicos requeridos para el buen funcionamiento de las sesiones de videoconferencia. La Tabla 9.4 detalla los requisitos funcionales que el sistema debe cumplir para proporcionar una gestión efectiva y eficiente de estos recursos, facilitando así la planificación y ejecución sin contratiempos de actividades educativas y administrativas.

Identificador	Descripción	Prioridad
RF-GR01	El sistema debe verificar la capacidad disponible (licencias "Large Meeting / Webinar" de Zoom o equivalentes en Teams) antes de confirmar la creación o modificación de una videoconferencia.	Alta
RF-GR02	El sistema debe balancear la capacidad de salas, creando o reasignando sesiones espejo cuando la matrícula supere la capacidad individual de la plataforma seleccionada.	Alta
RF-GR03	El sistema debe permitir a los administradores reasignar manualmente una videoconferencia a otra sala o licencia cuando se presenten conflictos de aforo.	Media
RF-GR04	El sistema debe generar reportes de uso de licencias y concurrencia por sala para planificación y escalabilidad.	Media

Tabla 9.4: Requisitos Funcionales de Gestión de Recursos (capacidad y licencias)

Nota: Esta tabla especifica los requisitos necesarios para la visualización y gestión de recursos dentro del sistema de videoconferencias. Prioridad media ha sido asignada a estos requisitos para reflejar su importancia en la optimización del uso de recursos y la planificación de actividades, garantizando que los recursos sean empleados de manera eficiente y estén alineados con las necesidades de los usuarios.

9.3.1.5. Orquestación de Aforo (OA)

En un periodo académico típico de la UNAD en cursos de primera matricula, puede superar los 27 000 estudiantes; sin embargo, las plataformas de videoconferencia imponen límites de participación interactiva que varían según la licencia:

- *Microsoft Teams Meeting* : 1 000 asistentes interactivos + 10 000 adicionales en modo *view-only*[1].
- *Zoom Meeting + Large-Meeting add-on* : 1 000 interactivos.
- *Zoom Webinar / Teams Town Hall* : hasta 10 000–20 000 asistentes solo-vista, con un panel limitado de presentadores interactivos.

Para impartir la misma clase en directo a toda la audiencia (sin saturar ninguna sala y manteniendo la interacción) el sistema creará **salas simultáneas interconectadas** (o *salas espejo*), distribuirá los estudiantes y unificará chats y preguntas.

ID	Requisito funcional	Prior.
RF-OA01	El sistema debe crear dinámicamente salas simultáneas interconectadas cuando la matrícula exceda la *capacidad interactiva configurada* de la plataforma/licencia elegida (p. ej. 1 000 en Teams Meeting).	Alta
RF-OA02	El sistema debe distribuir automáticamente a los estudiantes entre las salas, equilibrando la carga y evitando sobre-aforo.	Alta
RF-OA03	Antes de confirmar la clase, el sistema debe verificar y reservar la licencia adecuada (Zoom Large Meeting/Webinar o Teams Advanced Comms) y registrar su consumo.	Alta
RF-OA04	El sistema debe sincronizar contenido (vídeo, presentaciones, encuestas y grabaciones) entre la sala principal y las réplicas para garantizar una experiencia homogénea.	Media
RF-OA05	El sistema debe consolidar preguntas y chat procedentes de todas las salas y mostrarlas al docente en un panel unificado, permitiendo respuesta en vivo.	Media

Tabla 9.5: Requisitos funcionales del bloque *Orquestación de Aforo*

Nota. Este bloque implementa la lógica de *sharding* de audiencia y la gestión de licencias de gran aforo. Los requisitos de prioridad *Alta* garantizan la entrega de la clase a toda la matrícula; los de prioridad *Media* mejoran la interacción masiva (convergencia de chats, encuestas y grabaciones).

El diagrama de caso de uso asociado se presenta en la subsec:cu-oa, donde se modelan las interacciones entre el docente, el administrador del sistema y las API de Zoom/Teams que satisfacen estos requisitos.

[1]Microsoft, *Overview of meetings, webinars, and town halls*, https://learn.microsoft.com/en-us/microsoftteams/overview-meetings-webinars-town-halls, consultado junio 2025.

9.3.1.6. Informes y Monitoreo

El monitoreo continuo y la generación de informes son fundamentales para mantener y mejorar la calidad y eficacia de un sistema de gestión de videoconferencias. Estos procesos permiten a los administradores del sistema obtener información valiosa sobre el uso, la asistencia y la calidad de las sesiones, además de recibir alertas sobre problemas técnicos o de rendimiento. La Tabla 9.6 detalla los requisitos necesarios para implementar estas funcionalidades críticas, asegurando que el sistema pueda responder proactivamente a las necesidades de los usuarios y mantener un estándar alto de servicio.

Identificador	Descripción	Prioridad
RF-IM01	El sistema debe generar informes periódicos sobre el uso del sistema de videoconferencias (número de sesiones, duración, etc.).	Alta
RF-IM02	El sistema debe proporcionar métricas de rendimiento (tiempos de carga, latencia, etc.) en las videoconferencias realizadas.	Media
RF-IM03	El sistema debe generar alertas cuando se detecten problemas en el rendimiento del sistema (caídas, latencias altas, etc.).	Media
RF-IM04	El sistema debe permitir la descarga de reportes en formatos estándar (PDF, CSV, etc.) para fines de análisis.	Baja

Tabla 9.6: Requisitos Funcionales de Informes y Monitoreo

Nota: Esta tabla presenta los requisitos para el sistema de informes y monitoreo del sistema de videoconferencias, destacando la importancia de la visibilidad y control en la gestión operativa. Los requisitos de prioridad media reflejan su rol esencial en la optimización del servicio y la intervención temprana en caso de incidentes técnicos, lo que es crucial para la continuidad y calidad del servicio educativo.

9.3.1.7. Integración con Plataformas Externas (Zoom y Teams):

La integración fluida con plataformas externas como Zoom y Microsoft Teams es clave para ampliar la funcionalidad y accesibilidad del sistema de gestión de videoconferencias. Esta sección define los requisitos necesarios para asegurar que el sistema pueda interactuar efectivamente con estas plataformas, facilitando desde la sincronización de calendarios hasta la gestión automática de la autenticación y el acceso a las grabaciones. La Tabla 9.7 detalla estos requisitos, subrayando la importancia de una integración robusta que permita a los usuarios aprovechar al máximo las funcionalidades de videoconferencia sin problemas de compatibilidad o usabilidad.

Identificador	Descripción	Prioridad
RF-IP01	El sistema debe permitir la integración con las APIs de Zoom y Teams para la creación y gestión automática de videoconferencias.	Alta
RF-IP02	El sistema debe sincronizar los calendarios de Zoom y Teams con el sistema de agendamiento para evitar conflictos de horarios.	Alta
RF-IP03	El sistema debe gestionar automáticamente los tokens de autenticación necesarios para mantener la conexión con Zoom y Teams.	Alta
RF-IP04	El sistema debe permitir la descarga de grabaciones de las videoconferencias realizadas en Zoom y Teams.	Media
RF-IP05	El sistema debe enviar notificaciones automáticas a los usuarios cuando se generen enlaces de Zoom o Teams para sus videoconferencias.	Alta

Tabla 9.7: Requisitos Funcionales de Autenticación y Seguridad

Nota: Esta tabla especifica los requisitos para una integración exitosa con las APIs de Zoom y Teams, destacando la alta prioridad de mantener una interoperabilidad sin fisuras que soporte la gestión de videoconferencias a gran escala. Los requisitos presentados abordan aspectos críticos como la sincronización de calendarios y la autenticación automática, esenciales para la operación eficiente y segura del sistema. La prioridad media para la descarga de grabaciones refleja su rol en complementar la funcionalidad de documentación y revisión post-evento.

9.3.1.8. Notificaciones y Alertas:

Este bloque reúne los requisitos funcionales que permiten mantener a los usuarios informados y al equipo de soporte alerta ante eventos relevantes del sistema. Incluye desde la configuración de preferencias de aviso hasta la escalada de alertas no atendidas. La Tabla 9.8 resume estos requisitos, indicando su prioridad dentro del sistema.

Identificador	Descripción	Prioridad
RF-NA01	El sistema debe permitir a los usuarios configurar sus preferencias de notificación (correo electrónico, aplicación móvil, SMS).	Alta
RF-NA02	El sistema debe enviar notificaciones automáticas a los participantes cuando se programe, modifique o cancele una videoconferencia.	Alta
RF-NA03	El sistema debe generar alertas en tiempo real a los administradores cuando se detecten fallas de servicio o incidentes de seguridad.	Alta
RF-NA04	El sistema debe permitir a los usuarios confirmar la recepción de notificaciones críticas y registrar dicha confirmación.	Media
RF-NA05	El sistema debe escalar las alertas no atendidas al siguiente nivel de soporte después de un tiempo configurado.	Media

Tabla 9.8: Requisitos Funcionales de Notificaciones y Alertas

Nota: Esta tabla recoge las funcionalidades necesarias para asegurar una comunicación eficaz entre la plataforma y sus usuarios. Los requisitos de alta prioridad garantizan avisos inmediatos sobre la planificación de videoconferencias y la detección de incidentes críticos, mientras que los de prioridad media completan el flujo con confirmaciones de lectura y escalamiento de alertas.

9.3.2. Diagramas de Casos de Uso

En este apartado se presentan los diagramas de casos de uso, herramientas esenciales para la especificación de los requerimientos funcionales del sistema. Estos diagramas describen las interacciones entre los actores (usuarios del sistema) y el sistema mismo, para ilustrar cómo se deberían ejecutar los procesos principales. Para el Sistema de Agendamiento Multi-Sala de Videoconferencias, los diagramas de caso de uso se desarrollan para cubrir áreas críticas de la funcionalidad y administración del sistema, asegurando que todos los procesos clave estén claramente definidos y entendidos.

En correspondencia con la clasificación de requisitos funcionales presentada en la Sección 3.2.1, a continuación se incluye un diagrama de caso de uso por cada bloque funcional *Gestión de Videoconferencias, Autenticación y Seguridad, Administración del Sistema, Gestión de Recursos, Informes y Monitoreo, Integración con Plataformas Externas y Notificaciones y Alertas*, manteniendo el mismo orden para facilitar la trazabilidad entre las tablas de requisitos y su representación gráfica.

9.3.2.1. Gestión de Videoconferencias

El caso de uso "Gestión de Videoconferencias" es fundamental en el Sistema de Agendamiento Multi-Sala de Videoconferencias, ya que permite organizar y administrar las sesiones de videoconferencia de manera efectiva. Esta funcionalidad es crítica porque asegura que solo los usuarios autorizados puedan acceder y manipular los ajustes necesarios para llevar a cabo reuniones virtuales eficientes. La capacidad de gestionar videoconferencias de forma eficaz es clave para mantener la fluidez de la comunicación, la personalización de las sesiones y el control de acceso a los recursos tecnológicos, elementos vitales en un entorno que cada vez depende más de tecnologías de comunicación avanzadas para su operación diaria.

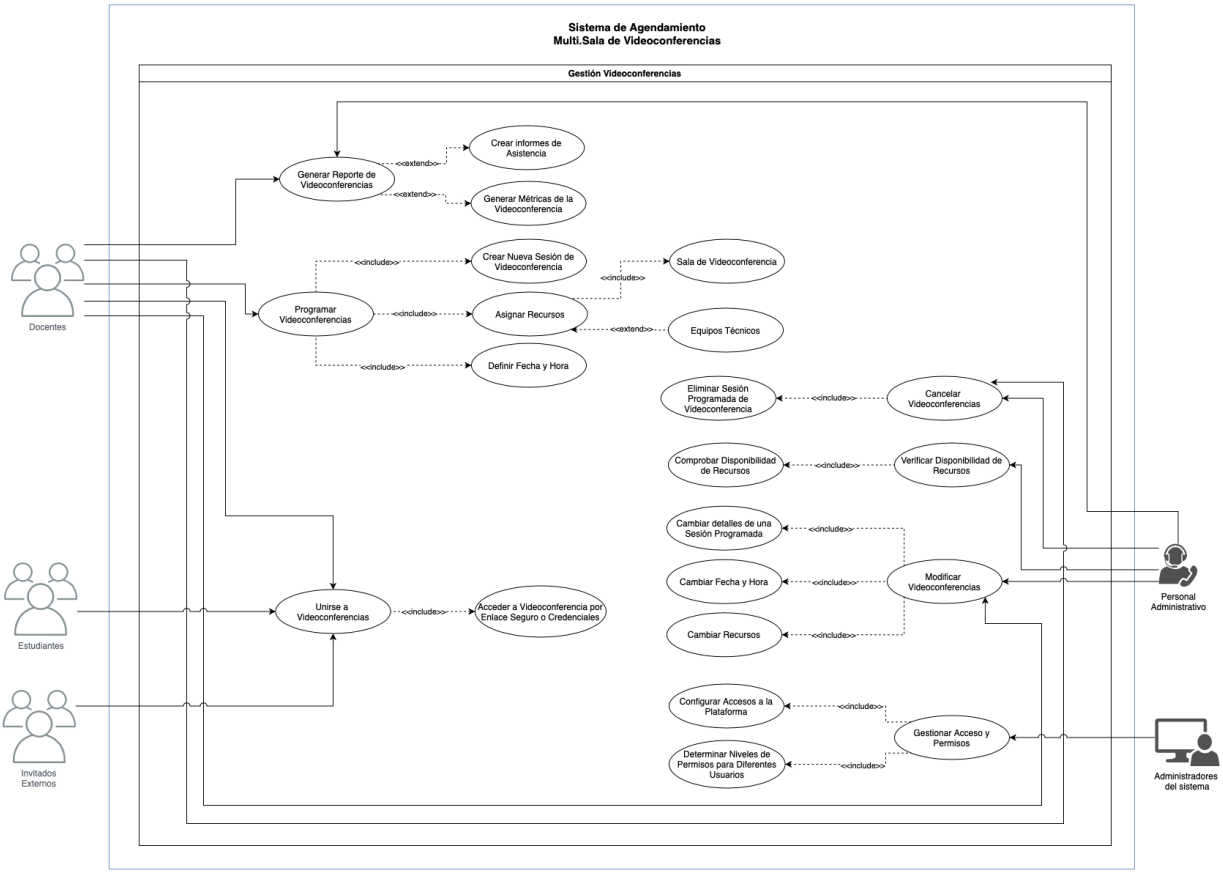


Figura 9.3: Caso de Uso Gestión de Videoconferencias del Sistema de Agendamiento Multi-Sala (Autor, 2024)

El diagrama presentado ilustra las operaciones principales que los administradores del sistema, docentes, estudiantes, invitados externos y personal administrativo pueden realizar dentro de la sección de gestión de videoconferencias del sistema de agendamiento de videoconferencias:

- *Crear Videoconferencia*
  Proceso que inicia con la programación de una nueva sesión de videoconferencia. Esto incluye la selección

de fecha, hora, y recursos necesarios. El administrador verifica la disponibilidad de los recursos y confirma la creación de la sesión, asegurando que todos los detalles sean correctos y que la sesión se ajuste a las necesidades de los usuarios.

- *Modificar Videoconferencia*
  Comienza con la selección de una sesión de videoconferencia existente. El administrador puede cambiar los detalles de la sesión, como la fecha, hora, y recursos asignados. Cada cambio es validado para asegurar su viabilidad antes de actualizar la información en el sistema. Este caso de uso es fundamental para adaptar las sesiones a cambios imprevistos o a nuevas necesidades de los participantes.

- *Cancelar Videoconferencia*
  Este proceso se inicia con la elección de una sesión programada para su cancelación. El administrador confirma la cancelación y procede a notificar a todos los participantes afectados, asegurando una comunicación eficiente y evitando inconvenientes.

- *Verificar Disponibilidad de Recursos*
  Antes de programar o modificar una videoconferencia, se verifica la disponibilidad de los recursos necesarios. Este paso es crucial para prevenir la programación de sesiones que no se puedan llevar a cabo por falta de disponibilidad de equipos o espacios.

- *Generar Reportes de Videoconferencias*
  El sistema permite generar diversos informes sobre las sesiones realizadas, incluyendo asistencia, uso de recursos, y feedback de los participantes. Estos informes son vitales para la mejora continua del servicio de videoconferencias.

- *Gestionar Acceso y Permisos*
  Se maneja la configuración de acceso a las videoconferencias, asegurando que solo los participantes autorizados puedan unirse. Además, se establecen los niveles de permisos para diferentes roles de usuarios, como moderadores o espectadores, para mantener la seguridad y orden durante las sesiones.

- *Unirse a Videoconferencias*
  Este caso de uso describe la acción de los participantes (estudiantes, docentes, e invitados externos) de acceder a las videoconferencias programadas. Comienza con la autenticación del usuario o la verificación de un enlace seguro proporcionado para la sesión. Una vez autenticados o verificados, los usuarios pueden unirse a la sesión en la fecha y hora establecidas. Este proceso es esencial para garantizar que solo los usuarios autorizados y con las credenciales correctas puedan acceder a las videoconferencias, manteniendo así la privacidad y seguridad de las sesiones.

9.3.2.2. Autenticación y Seguridad

El caso de uso *Autenticación y Seguridad* integra los mecanismos que protegen la plataforma de videoconferencias contra accesos no autorizados y garantizan la confidencialidad de las comunicaciones. Abarca desde la verificación de credenciales institucionales hasta la administración de roles y permisos, y satisface los requisitos **RF-AS01** a **RF-AS08**.

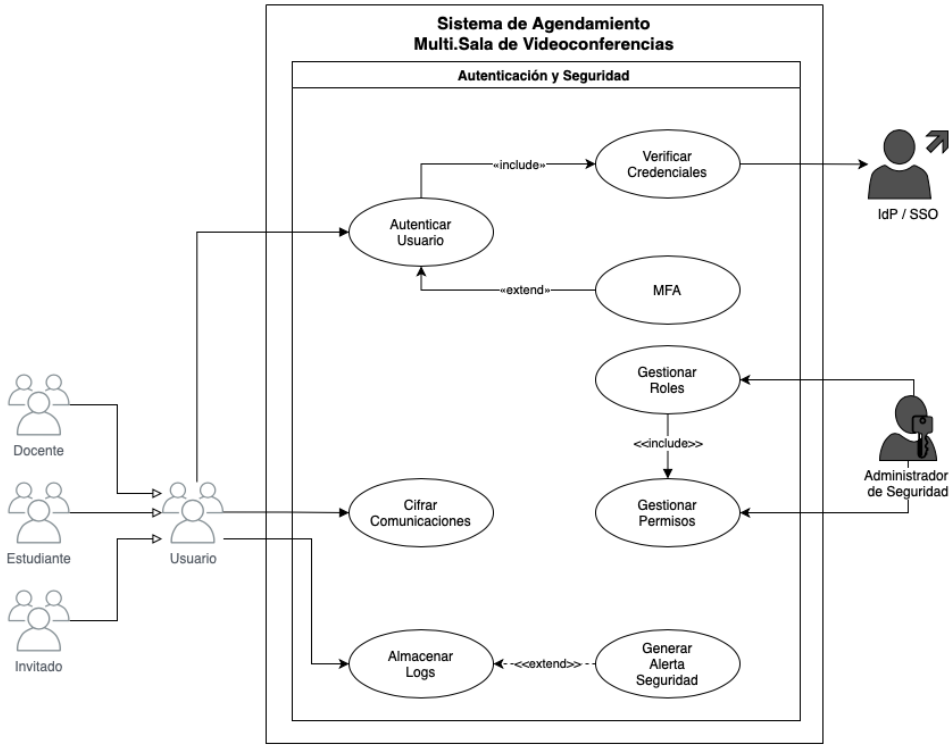


Figura 9.4: Caso de Uso de Autenticación y Seguridad del Sistema de Agendamiento Multi-Sala (Autor, 2025)

El diagrama ilustra las operaciones principales que los usuarios y el equipo de ciberseguridad pueden realizar dentro de la sección de autenticación y protección del sistema:

- *Autenticar Usuario*
  Inicia la sesión validando las credenciales institucionales del usuario ante el *IdP / SSO* (Identity Provider / Single Sign-On). Incluye, de forma obligatoria, la verificación de credenciales y, cuando la política lo exige, se extiende con *MFA* (Multi-Factor Authentication).

- *Verificar Credenciales*
  Confirma usuario y contraseña mediante protocolos seguros OAuth 2.0 (Open Authorization) o SAML 2.0 (Security Assertion Markup Language) (RF-AS03).

- *MFA*
  Añade un segundo factor de autenticación —código *TOTP* (Time-based One-Time Password) o llave

FIDO2 (Fast Identity Online 2)— para los roles que lo requieran (RF-AS04).

- *Cifrar Comunicaciones*
  Garantiza el uso de *TLS 1.3* (Transport Layer Security v1.3) en cada sesión, protegiendo datos en tránsito y en reposo (RF-AS05).

- *Gestionar Roles*
  Permite al *Administrador de Seguridad* crear y modificar roles. Incluye *Gestionar Permisos* para definir privilegios de acuerdo con el modelo *RBAC* (Role-Based Access Control) (RF-AS08).

- *Gestionar Permisos*
  Asigna o revoca permisos específicos dentro del modelo RBAC.

- *Almacenar Logs*
  Registra actividades relevantes para auditoría y cumplimiento (RF-AS06).

- *Generar Alerta Seguridad*
  Se activa como extensión cuando se detectan intentos de acceso fallidos o acciones no autorizadas (RF-AS07).

Las relaciones «*include*» señalan pasos imprescindibles dentro del flujo (por ejemplo, toda autenticación verifica credenciales), mientras que las relaciones «*extend*» representan comportamientos opcionales que se desencadenan bajo condiciones específicas, como la exigencia de MFA o la generación de alertas de seguridad.

9.3.2.3. Administración del Sistema

El caso de uso *Administración del Sistema* reúne las actividades que permiten a los administradores controlar la operación diaria de la plataforma, garantizar la disponibilidad de recursos y ajustar la configuración conforme a las políticas institucionales. Este paquete satisface los requisitos **RF-AD01** a **RF-AD07** y sirve de punto central para la gestión operativa y de seguridad del servicio de videoconferencias.

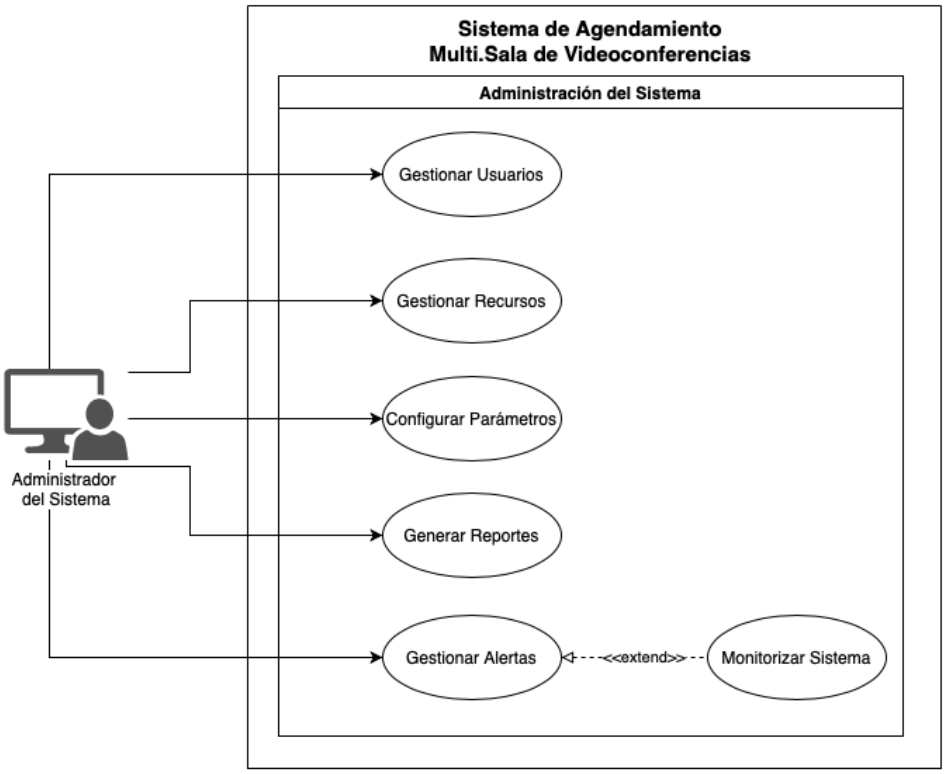


Figura 9.5: Caso de Uso de Administración del Sistema del Sistema de Agendamiento Multi-Sala (Autor, 2025)

El diagrama muestra las operaciones principales disponibles para el *Administrador del Sistema*:

- *Gestionar Usuarios* (RF-AD01)
  Altas, bajas y modificaciones de cuentas. El flujo detallado se presenta en el Anexo A.

- *Monitorizar Sistema* (RF-AD02)
  Panel en tiempo real con métricas de uso (sesiones activas, carga, ancho de banda).

- *Gestionar Recursos* (RF-AD03)
  Asignación y optimización de salas, perfiles de ancho de banda y licencias disponibles.

- *Gestionar Alertas* (RF-AD04)
  Registro y notificación de incidencias técnicas. Se activa como *extensión* de *Monitorizar Sistema* cuando

se detectan valores críticos.

- *Generar Reportes* (RF-AD05)
  Elaboración de informes de uso (número de videoconferencias, duración, participantes) en PDF/CSV para auditoría y mejora continua.

- *Configurar Parámetros* (RF-AD06)
  Ajuste de políticas (límites de sesión, políticas de grabación, valores de umbral para alertas) según la institución.

- *Auditoría de Actividades* (RF-AD07)
  Este requisito se implementa transversalmente mediante los registros generados por *Gestionar Usuarios*, *Generar Reportes* y *Gestionar Alertas*, asegurando trazabilidad completa.

La relación *Monitorizar Sistema* → *Gestionar Alertas* se modela como «extend»: el manejo de alertas solo se ejecuta cuando el monitoreo detecta una situación anómala que lo justifique. Así, el diagrama refleja la dependencia condicional entre la supervisión continua y la respuesta a incidentes.

9.3.2.4. Gestión de Recursos

El caso de uso *Gestión de Recursos* garantiza que las salas de videoconferencia, el ancho de banda y los equipos técnicos se utilicen de manera óptima y estén disponibles cuando los usuarios programan sesiones. Esta funcionalidad cumple los requisitos **RF-GR01** a **RF-GR04**, proporcionando verificación de disponibilidad en tiempo real, asignación automática o manual de recursos y reportes de uso para la planificación de capacidad.

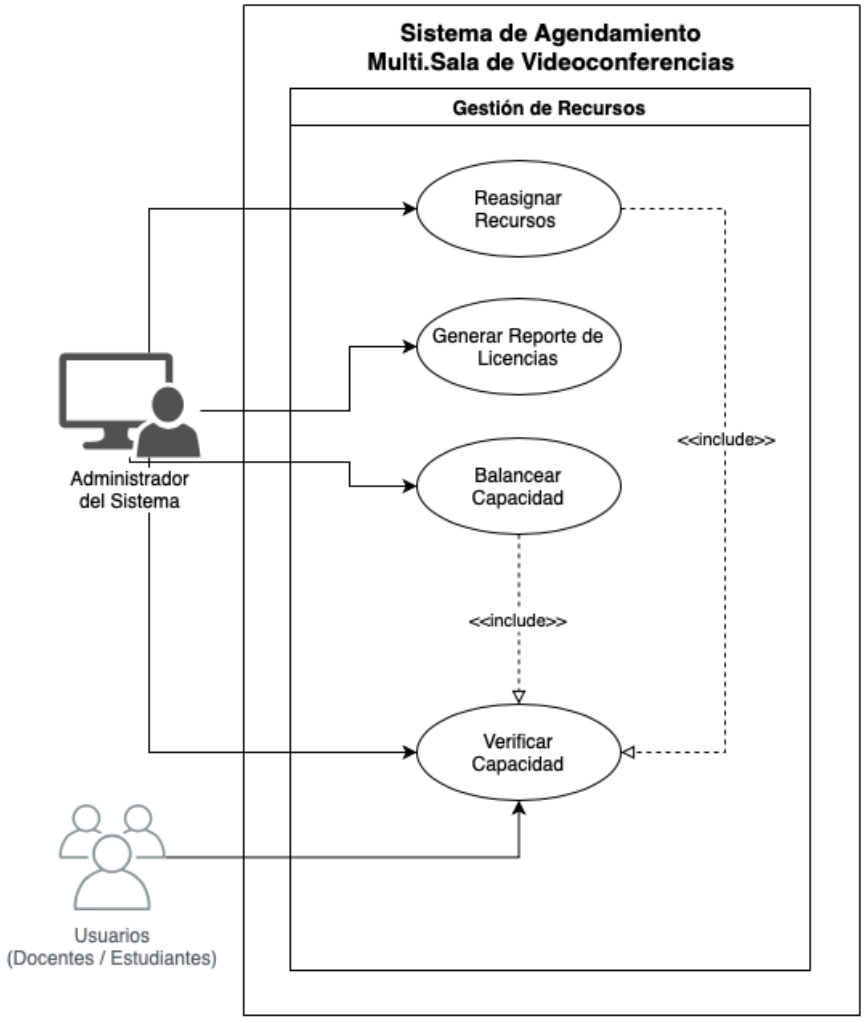


Figura 9.6: Caso de Uso de Gestión de Recursos del Sistema de Agendamiento Multi-Sala (Autor, 2025)

El diagrama ilustra las operaciones principales que puede realizar el *Administrador del Sistema*, así como la interacción indirecta de los docentes y estudiantes cuando programan videoconferencias:

- *Verificar Disponibilidad* (RF-GR01)
  Comprueba, antes de crear o modificar una sesión, si existen salas y ancho de banda suficientes. Esta verificación es obligatoria y se ejecuta cada vez que cualquier actor agenda una videoconferencia.

- *Optimizar Uso de Recursos* (RF-GR02)
  Reasigna automáticamente videoconferencias a la sala o enlace con mejor capacidad, reduciendo cuellos de botella. Incluye *Verificar Disponibilidad* para validar los cambios propuestos.

- *Asignar Recursos Manualmente* (RF-GR03)
  Permite al administrador forzar o reubicar una sesión cuando existe un conflicto de capacidad o se requiere soporte especial. También *incluye* la verificación previa de disponibilidad.

- *Generar Reporte de Recursos* (RF-GR04)
  Produce informes periódicos de ocupación de salas y consumo de ancho de banda, facilitando la planificación y escalabilidad de la infraestructura.

Las relaciones «*include*» reflejan que tanto la optimización automática como la asignación manual dependen necesariamente de la verificación de disponibilidad, asegurando coherencia y consistencia en el uso de los recursos.

9.3.2.5. Orquestación de Aforo

El caso de uso *Orquestación de Aforo* coordina la entrega de una misma clase en directo a cohortes que superan la capacidad interactiva de Zoom o Microsoft Teams.[2] Para garantizar que cada estudiante pueda participar (chat, preguntas y reacciones) sin saturar una sola sala, el sistema crea sesiones simultáneas interconectadas (*salas espejo*), reparte la audiencia y sincroniza el contenido con la plataforma seleccionada.

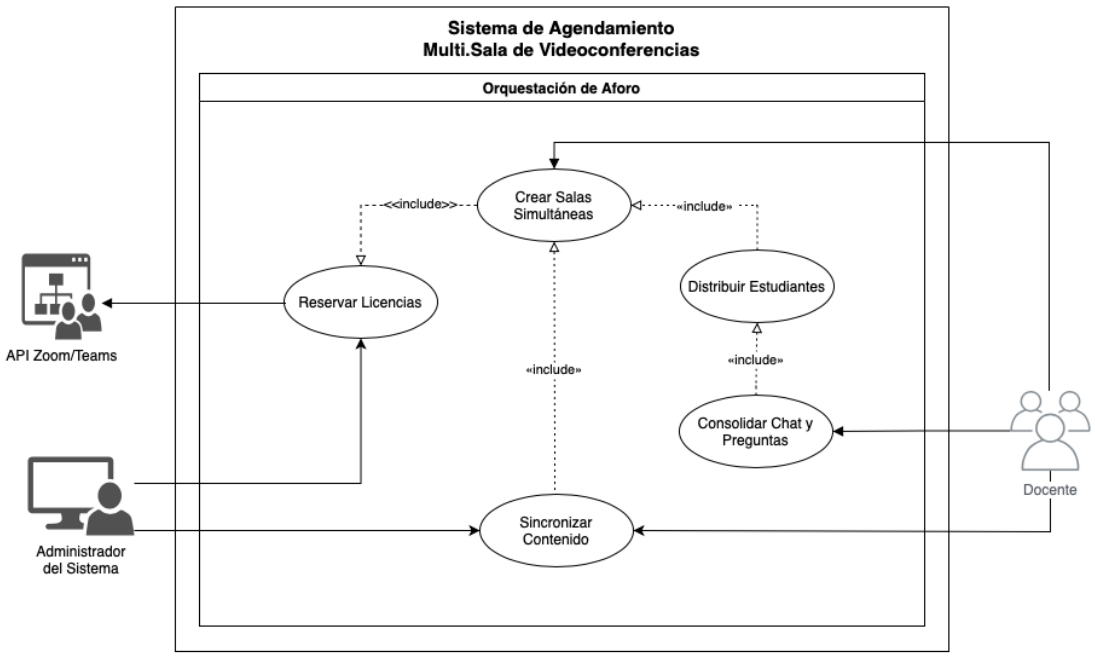


Figura 9.7: Caso de Uso *Orquestación de Aforo* del Sistema de Agendamiento Multi-Sala (Autor, 2025)

El diagrama modela las interacciones entre el Docente, el Administrador del Sistema y la API Zoom/-Teams para satisfacer los requisitos RF-OA01 – RF-OA05:

- *Reservar Licencias* (RF-OA03)
  Solicita a la API las licencias *Large Meeting/Webinar* o *Advanced Communications*. Es un paso imprescindible antes de crear salas espejo, por ello se *incluye* en el flujo de *Crear Salas Simultáneas*.

- *Crear Salas Simultáneas* (RF-OA01)
  Genera automáticamente las salas espejo cuando la matrícula supera el aforo permitido. Es el caso de uso base del bloque.

- *Distribuir Estudiantes* (RF-OA02)
  Reparte la audiencia de forma equilibrada entre las salas espejo evitando sobrecarga. Se activa siempre que se crean salas adicionales (*include*).

[2]Por ejemplo, Teams Meeting admite 1 000 asistentes interactivos más 10 000 en modo *view-only*; Zoom Meeting con el *Large-Meeting add-on* llega a 1 000; los Webinar/Town Hall amplían el aforo pero restringen la interacción.

- *Sincronizar Contenido* (RF-OA04)
  Replica presentación, grabación y encuestas en cada sala. Incluido desde *Crear Salas* para asegurar coherencia en tiempo real.

- *Consolidar Chat y Preguntas* (RF-OA05)
  Recoge los mensajes de todas las salas espejo y los muestra al docente en un panel unificado, garantizando igualdad de participación. Depende de la distribución de estudiantes, por lo que también se marca como *include*.

Las relaciones «*include*» reflejan pasos obligatorios: *Crear Salas* no puede finalizar sin licencias adecuadas, copia de contenido ni balance de estudiantes; a su vez, agrupar el chat requiere que la audiencia ya esté distribuida. Con esta orquestación se logra impartir la clase a más de 27 000 estudiantes manteniendo la interacción y la calidad del servicio.

9.3.2.6. Informes y Monitoreo

El caso de uso *Informes y Monitoreo* agrupa las funcionalidades que permiten a los operadores y técnicos supervisar el funcionamiento del Sistema de Agendamiento Multi-Sala de Videoconferencias. Esta capacidad resulta esencial para mantener la calidad del servicio, detectar tempranamente incidencias de rendimiento y disponer de información histórica que facilite la toma de decisiones académicas y técnicas. Además, la trazabilidad de los tickets de soporte asegura que cada problema sea gestionado y documentado de manera adecuada, cumpliendo con el requerimiento no funcional de un **MTTR (Mean Time To Recovery)** promedio mensual menor o igual a 15 minutos.

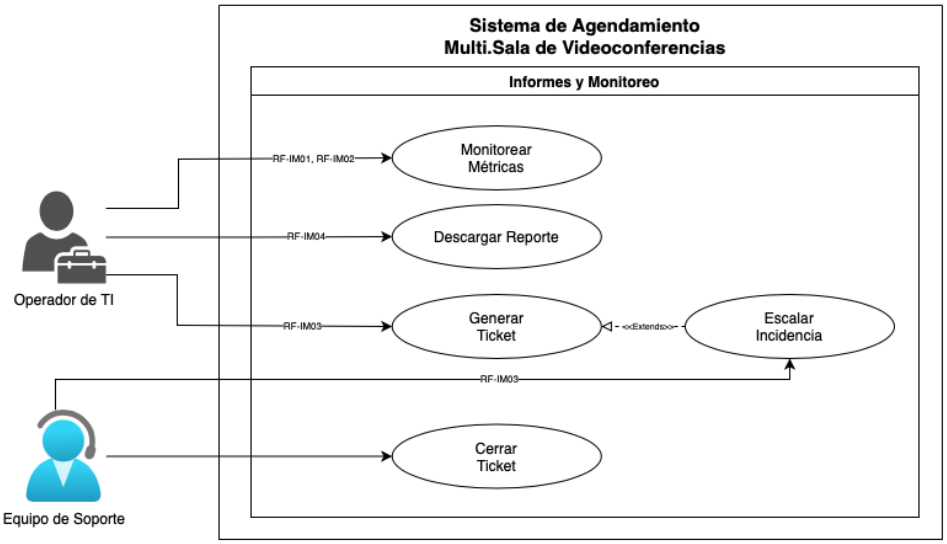


Figura 9.8: Caso de Uso de Informes y Monitoreo del Sistema de Agendamiento Multi-Sala (Autor, 2025)

El diagrama presentado ilustra las operaciones principales que los operadores y técnicos de soporte pueden realizar para garantizar la continuidad y la eficiencia del servicio de videoconferencias:

- *Monitorear Métricas*
  Consulta en tiempo real indicadores clave de rendimiento (número de sesiones, latencia, uso de recursos). Cubre los requisitos RF-IM01 y RF-IM02, activando umbrales de alerta cuando se detectan valores anómalos.

- *Descargar Reporte*
  Genera informes periódicos y permite su descarga en formatos estándar (PDF, CSV). Este caso de uso satisface el requisito RF-IM04 y facilita el análisis histórico por parte de las unidades académicas.

- *Generar Ticket*
  Registra una incidencia cuando se identifica un problema de rendimiento o disponibilidad. Implementa el requisito RF-IM03 y da inicio al ciclo de vida del soporte.

- *Escalar Incidencia*
  Extiende a *Generar Ticket* cuando la gravedad del problema supera el umbral definido; transfiere el ticket a un nivel superior de soporte o a proveedores externos. Mantiene la trazabilidad y el cumplimiento del MTTR especificado.

- *Cerrar Ticket*
  Finaliza la incidencia una vez aplicada y verificada la solución, actualizando el registro histórico y garantizando la retroalimentación hacia los procesos de mejora continua.

El diagrama también destaca el uso de relaciones de "*extend*" para reflejar la escalabilidad del proceso de soporte: *Escalar Incidencia* sólo se ejecuta en aquellos casos donde la solución supera las competencias del primer nivel de atención.

9.3.2.7. Integración con Plataformas Externas

El caso de uso *Integración con Plataformas Externas* conecta el Sistema de Agendamiento Multi-Sala con las API oficiales de Zoom y Microsoft Teams para automatizar la gestión de reuniones, la autenticación delegada y la sincronización de contenido. De esta forma se satisfacen los requisitos RF-IP01–RF-IP05, garantizando interoperabilidad sin fricciones y evitando tareas manuales de los docentes y del equipo de TI.

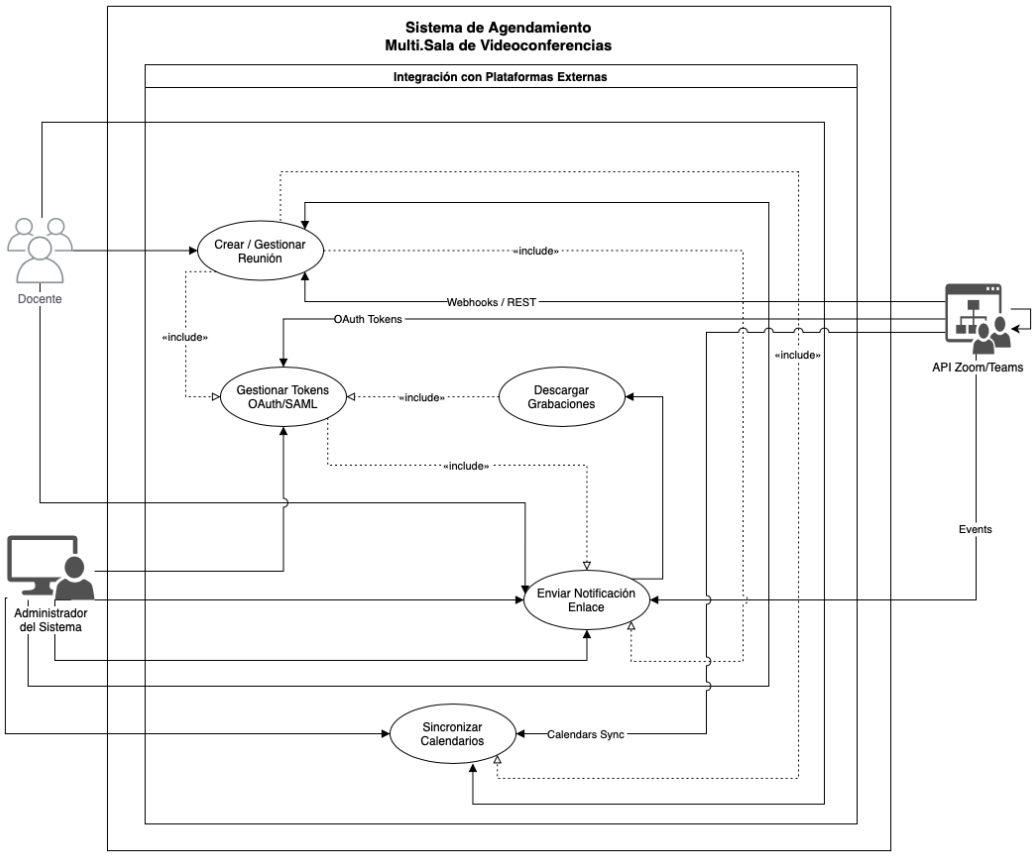


Figura 9.9: Caso de Uso Integración con Plataformas Externas (Autor, 2025)

El diagrama muestra las principales operaciones y su relación con las API de Zoom/Teams (expuestas por *REST* y *Webhooks*):

- *Crear / Gestionar Reunión*
  Permite al docente programar, modificar o cancelar sesiones directamente desde el sistema. El flujo usa los `endpoints` de reuniones y desencadena *webhooks* de confirmación (RF-IP01).

- *Gestionar Tokens OAuth/SAML*
  Tras vincular (federar) el directorio institucional con Zoom/Teams, el sistema guarda de forma segura los *refresh tokens* emitidos por el IdP y los renueva automáticamente; así puede invocar las APIs externas sin que ningún usuario tenga que volver a introducir sus credenciales (RF-IP03).

- *Descargar Grabaciones*
  Tras finalizar la sesión, el sistema recupera las grabaciones disponibles en la nube de Zoom/Teams y las enlaza al curso virtual (RF-IP04).

- *Sincronizar Calendarios*
  Mantiene coherentes los calendarios institucionales y los de videoconferencia; se ejecuta en segundo plano cada vez que cambian la fecha u hora de la reunión (RF-IP02).

- *Enviar Notificación Enlace*
  Publica el URL definitivo de la sala (o sus cambios) en los canales de notificación del LMS y correo institucional, cumpliendo RF-IP05.

Las relaciones «*include*» indican pasos obligatorios dentro del flujo (p. ej. toda reunión requiere la gestión de tokens), mientras que las asociaciones sólidas modelan los mensajes REST/Webhook intercambiados con las plataformas externas.

9.3.2.8. Notificaciones y Alertas

El caso de uso *Notificaciones y Alertas* centraliza la mensajería operativa del Sistema de Agendamiento Multi-Sala de Videoconferencias. Su objetivo es mantener informados a los participantes y al equipo de soporte sobre eventos relevantes (desde recordatorios de videoconferencias hasta incidentes de servicio) cumpliendo los requisitos **RF-NA01** a **RF-NA05**.

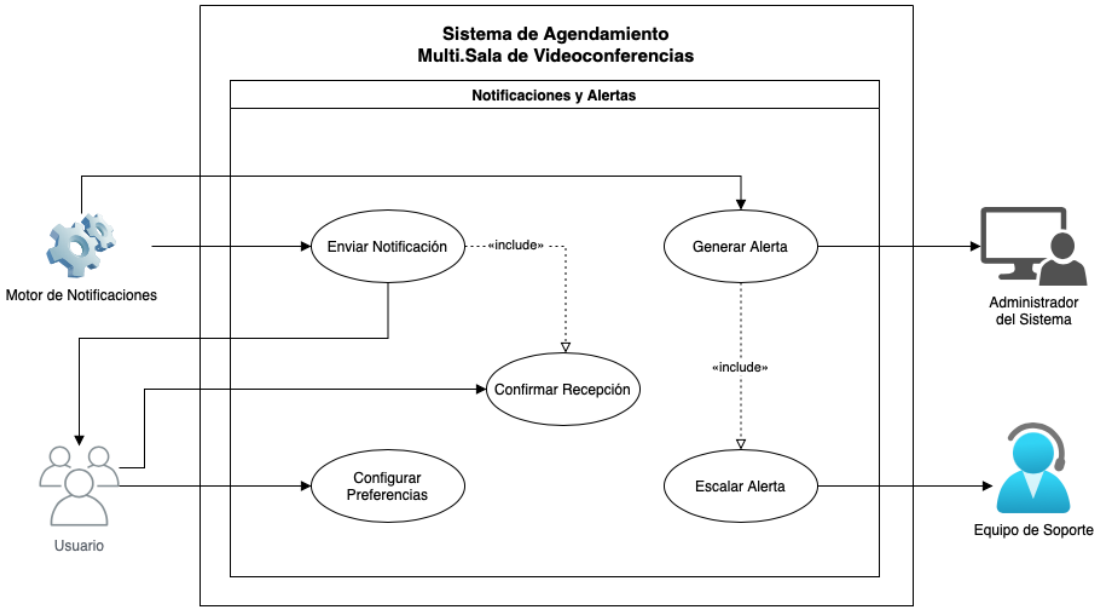


Figura 9.10: Caso de Uso de Notificaciones y Alertas del Sistema de Agendamiento Multi-Sala (Autor, 2025)

El diagrama refleja las interacciones entre el *Motor de Notificaciones* (servicio interno que orquesta los envíos), los usuarios finales, el equipo de soporte y el administrador del sistema:

- *Configurar Preferencias*
  Cada usuario define cómo desea recibir avisos (correo, aplicación móvil, SMS) y cuáles eventos son relevantes (RF-NA01).

- *Enviar Notificación*
  El motor entrega mensajes automáticos cuando se crea, modifica o cancela una videoconferencia (RF-NA02). Incluye *Confirmar Recepción* para los avisos que exigen acuse.

- *Confirmar Recepción*
  El usuario marca la notificación como leída; el sistema registra la confirmación para auditoría (RF-NA04).

- *Generar Alerta*
  Ante fallas de servicio o eventos críticos, el motor crea una alerta en tiempo real y la notifica al administrador (RF-NA03). Puede *incluir Escalar Alerta*.

- *Escalar Alerta*
  Si la alerta no es atendida dentro del plazo configurado, se reenvía al siguiente nivel de soporte o a un grupo de guardia, garantizando la continuidad operativa (RF-NA05).

Las relaciones «*include*» explicitan pasos indispensables (enviar → confirmar, generar → escalar), mientras que el *Motor de Notificaciones* se modela como un servicio interno que desencadena los casos de uso sin intervención humana.

9.3.2.9. Administración de Usuarios

El caso de uso *Administración de Usuarios* es fundamental en el Sistema de Agendamiento Multi-Sala de Videoconferencias, ya que permite gestionar de manera eficiente las cuentas de usuario dentro del sistema. Esta funcionalidad es crítica porque asegura que solo los usuarios autorizados puedan acceder y realizar operaciones según los roles y permisos asignados. La capacidad de administrar usuarios efectivamente es clave para mantener la seguridad, la personalización y el control de acceso al sistema, elementos vitales en un entorno educativo que depende cada vez más de tecnologías de información para su operación diaria.

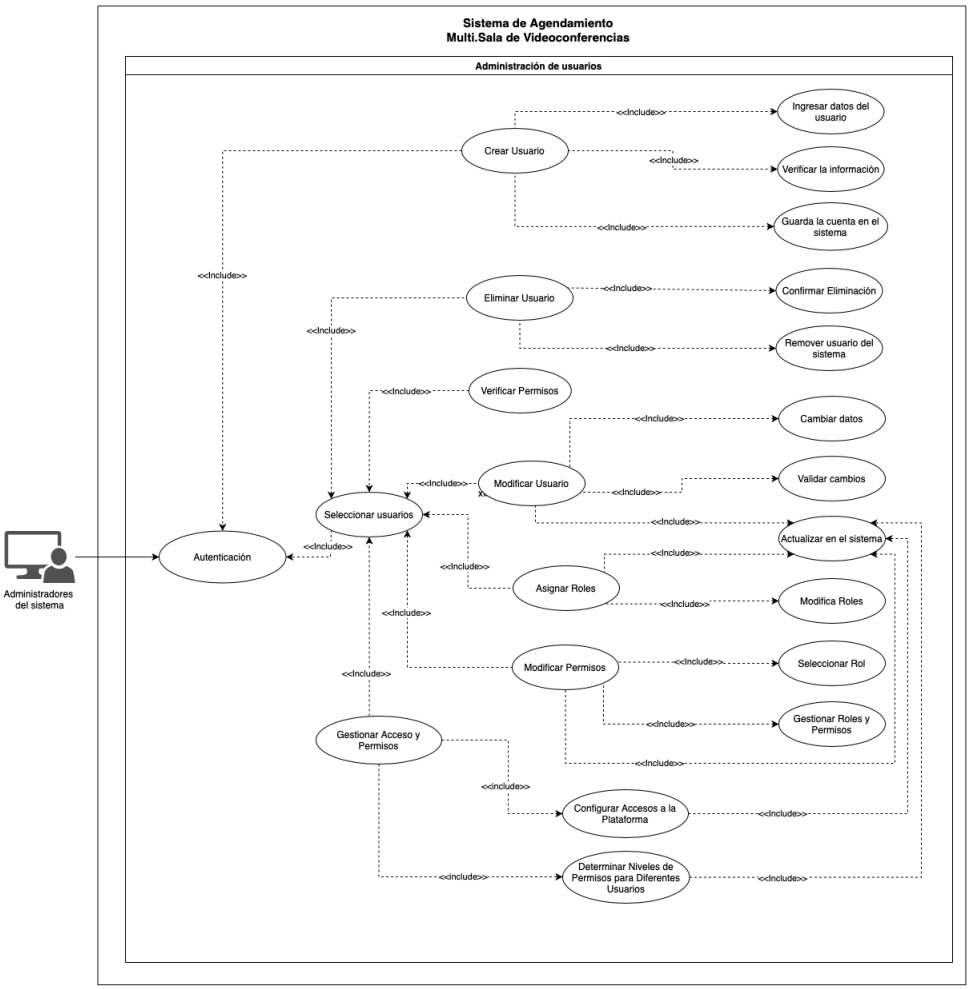


Figura 9.11: Caso de Uso de Administración de Usuarios del Sistema de Agendamiento Multi-Sala (Autor, 2024)

El diagrama presentado ilustra las operaciones principales que los administradores del sistema pueden realizar dentro de la sección de administración de usuarios del sistema de agendamiento de videoconferencias:

- *Autenticación*
  La autenticación es el proceso inicial que permite al administrador del sistema acceder a las funciones de administración de usuarios. Este paso asegura que solo personal autorizado tenga acceso al sistema, estableciendo una primera capa de seguridad.

- *Verificar Permisos*
  Después de la autenticación, el sistema realiza una verificación de permisos cada vez que se selecciona un usuario para alguna operación. Este paso es esencial para validar que el administrador tiene los derechos necesarios para realizar acciones críticas, como modificar o eliminar cuentas de usuario.

- *Crear Usuario*
  Proceso que inicia con el ingreso de datos del usuario, seguido de la verificación de esta información, y culmina con el almacenamiento de la cuenta en el sistema. Este caso de uso incluye tres subprocesos que son esenciales para asegurar que los datos ingresados sean correctos y que la cuenta creada cumpla con todas las normativas y requerimientos del sistema.

- *Modificar Usuario*
  Inicia con la selección de un usuario existente, permitiendo cambiar sus datos, validar estos cambios y finalmente actualizarlos en el sistema. Este caso de uso garantiza que las cuentas de usuario puedan adaptarse a cambios en los roles o en la información personal de los usuarios.

- *Eliminar Usuario*
  Comienza también con la selección del usuario y procede con la confirmación de la eliminación antes de remover definitivamente la cuenta del sistema. Esto asegura que las bajas de usuarios sean manejadas de manera segura y controlada.

- *Asignar Roles*
  También inicia con la selección del usuario al cual se le asignará el rol, seguido el administrador selecciona el rol específico de un conjunto predeterminado, y el proceso culmina con la actualización de la cuenta del usuario en el sistema. Este caso de uso es esencial para la gestión de la seguridad y la distribución adecuada de responsabilidades dentro del sistema.

- *Modificar Permisos*
  Comienza también con la selección del usuario para posteriormente seguir con la elección del rol al que se le ajustarán los permisos. Luego, el administrador modifica los permisos asociados a ese rol, ya sea agregando nuevas capacidades o revocando permisos existentes, en función de las necesidades operativas o cambios en las políticas de seguridad. Finalmente, el proceso termina con la actualización de estos permisos en el sistema. Este caso de uso es crucial para mantener la integridad y la adecuación del sistema a las operaciones y políticas vigentes de la organización.

- *Gestionar Acceso y Permisos*
  Este caso de uso permite al administrador definir y configurar los niveles de acceso y permisos para diferentes usuarios en el sistema. Incluye subprocesos como *Configurar Acceso a la Plataforma* y *Determinar Niveles de Permisos para Diferentes Usuarios*, asegurando una distribución clara de las responsabilidades y capacidades de cada rol dentro del sistema.

Capítulo 10

Anexo B

Este apéndice contiene las tablas que documentan los pasos clave del análisis cualitativo: la categorización inicial de dificultades, su recategorización final en macrogrupos y las categorías refinadas definitivas. Estas tablas sirven de respaldo al apartado de Análisis cualitativo 4.5.2 y aportan la base empírica para extraer los atributos de calidad descritos en el capítulo.

Tabla 10.1: Tabla B.1: Categorización inicial de dificultades reportadas (insumo para atributos de calidad).

Categoría	**Descripción breve**	**Ejemplos de respuestas asociadas**
Conectividad / Ancho de banda	Problemas de conexión, cortes o baja calidad de transmisión.	"Fallas de conectividad", "la señal no es estable", "bajo ancho de banda"
Limitaciones técnicas / Dispositivos	Falta de equipos adecuados (cámaras, micrófonos).	"No todos tienen cámara", "fallas en micrófono"
Capacidad de la plataforma	Restricciones en número de usuarios o tiempo de reunión.	"Zoom se bloquea con muchos usuarios", "límite de tiempo free"
Falta de acompañamiento institucional	Carencia de soporte técnico o capacitación formal.	"No hubo inducción", "sin acompañamiento técnico"
Interfaz confusa / Usabilidad	Navegación poco intuitiva, exceso de opciones.	"Muchos botones", "no sé cómo programar clases"
Falta de funciones específicas	Herramientas ausentes (asignación automática, grabaciones).	"No puedo automatizar salas", "falta control de asistencia"
Inestabilidad de la plataforma	Errores, cierres inesperados o congelamientos.	"Zoom se cierra solo", "Teams se congela"
Falta de adaptación pedagógica	Dificultad para trasladar estrategias didácticas.	"Es difícil mantener la atención", "no se pueden hacer talleres"
Acceso desigual / Brecha digital	Usuarios sin condiciones mínimas para conectarse.	"Estudiantes sin internet en zonas rurales"
Otros / No especificado	Respuestas ambiguas o en blanco.	"No aplica", "ninguna dificultad"

Tabla 10.2: Tabla B.2: Recategorización final de dificultades (categorías simplificadas).

Categoría simplificada	**Equivalencias originales**
Conectividad / Ancho de banda	Igual (se mantiene)
Acceso tecnológico	Fusión de "Limitaciones técnicas" y "Brecha digital"
Acompañamiento institucional	Igual (se mantiene)
Funcionalidad y usabilidad	Fusión de "Interfaz confusa" y "Funciones faltantes"
Rendimiento y estabilidad	Fusión de "Inestabilidad" y "Capacidad" (opcional)
Adaptación pedagógica	Igual (se mantiene)
Otros / No especificado	Igual (se mantiene)

Tabla 10.3: Tabla B.3: Categorías refinadas de dificultades en el uso de videoconferencias.

Categoría	**Descripción breve**	**Ejemplos de respuestas asociadas**	**Ítems**
Conectividad y acceso	Internet inestable, falta de dispositivos.	“Fallas de conexión”, “no todos tienen cámara”	1,5,12,16,23
Problemas técnicos e inestabilidad	Fallos de la aplicación, cierres o congelamientos.	“Zoom se cierra solo”, “Teams se congela”	4,7,14,18,31
Interfaz y usabilidad	Navegación confusa, exceso de elementos.	“Muchos botones”, “difícil programar”	6,9,15,25,27
Funciones clave faltantes	Control de asistencia, salas automáticas.	“No puedo automatizar grupos”	2,13,19,21,26
Acompañamiento institucional	Falta de capacitación o soporte.	“No hubo inducción”	8,10,22,24,28
Adaptación pedagógica	Dificultad para estrategias didácticas.	“Difícil hacer talleres”	3,11,17,20,29
Coordinación y logística	Gestión de horarios y enlaces.	“Problemas con los enlaces”	30
Otros / No especificado	Respuestas vagas o en blanco.	“No aplica”	—

Capítulo 11

Anexo C

Población de referencia.
El **95 %** del estudiantado se conecta desde territorio colombiano (*analytics* de la plataforma institucional, 2024–2025), por lo que resulta metodológicamente válido presentar únicamente las mediciones realizadas en Cali (CO) como caso representativo.

Procedimiento. 25 peticiones HTTP por región a `https://dynamodb.<region>.amazonaws.com` (herramienta CloudPingTest [216]); se reporta la latencia RTT media y su dispersión.

Tabla 11.1: Latencia media desde Cali (25 rondas por región)

Región AWS	n	μ (ms)	σ	Mín–Máx
`us-east-1`	25	121.4	55.7	77–228
`sa-east-1`	25	199.9	70.2	134–413
`us-west-2`	25	315.6	82.1	260–437
`eu-west-1`	25	640.5	45.3	596–781

Resultado. Las regiones `us-east-1` (N. Virginia) y `sa-east-1` (São Paulo) ofrecen la menor latencia percibida (< 200 ms), cumpliendo así el margen definido en el escenario ATAM de Escalabilidad (Cap. 5, § 4.4).

Bibliografía

[1] W. Bao. «COVID-19 and Online Teaching in Higher Education: A Case Study of Peking University». En: *Human Behavior and Emerging Technologies* 2 (2020), págs. 113-115. DOI: 10.1002/hbe2.191.

[2] S. Dhawan. «Online Learning: A Panacea in the Time of COVID-19 Crisis». En: *Journal of Educational Technology Systems* 49 (2020), págs. 5-22. DOI: 10.1177/0047239520934018.

[3] P. R. Lowenthal y J. C. Dunlap. «The Role of the Instructor in the Transition to Online Learning: Lessons Learned from COVID-19». En: *Online Learning* 24 (2020), págs. 1-4. DOI: 10.24059/olj.v24i2.2238.

[4] V. J. García-Morales, A. Garrido-Moreno y R. Martín-Rojas. «The transformation of higher education after the COVID disruption: Emerging challenges in an online learning scenario». En: *Frontiers in Psychology* 12 (2021), pág. 616059. DOI: 10.3389/fpsyg.2021.616059.

[5] J. W. Creswell y V. L. P. Clark. *Designing and Conducting Mixed Methods Research*. 3.ª ed. Sage, 2017.

[6] P. Clements, R. Kazman y M. Klein. *Evaluating Software Architectures: Methods and Case Studies*. Boston, MA: Addison–Wesley, 2002. ISBN: 0-201-70482-X.

[7] R. Kazman, M. Klein y P. Clements. *The Architecture Tradeoff Analysis Method (ATAM) Version 2.0*. Inf. téc. CMU/SEI-2002-TR-028. Software Engineering Institute, Carnegie Mellon University, 2002.

[8] L. Bass, P. Clements y R. Kazman. «Understanding Quality Attributes». En: *Software Architecture in Practice*. 2.ª ed. Addison–Wesley, 2003. Cap. 4, págs. 69-104.

[9] L. Bass, P. Clements y R. Kazman. *Software Architecture in Practice*. 4th. Addison-Wesley, 2012.

[10] R. S. Pressman, I. Sommerville y D. Garlan. *Software Architecture: Foundations, Theory, and Practice*. Wiley, 2010. ISBN: 978-0-470-16774-8.

[11] H. Cervantes y R. Kazman. *Designing Software Architectures: A Practical Approach*. Addison–Wesley Professional, 2016. ISBN: 978-0-13-439078-9.

[12] G. Hohpe y B. Woolf. *Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions*. Addison–Wesley, 2003. ISBN: 978-0-321-20068-5.

[13] K. Deenadayalan y S. Menon. «Cloud Scalability Patterns and Practices: A Systematic Review». En: *Journal of Cloud Computing* 13.2 (2024), págs. 45-67. DOI: 10.1007/s13677-024-00458-9.

[14] *ISO/IEC 25010:2011 Systems and Software Engineering — Systems and Software Quality Requirements and Evaluation (SQuaRE) — System and Software Quality Models*. https://iso25000.com. Último acceso: 2 jul 2025. 2011.

[15] E. Hammer-Lahav, D. Hardt y J. Richer. *The OAuth 2.0 Authorization Framework*. Inf. téc. RFC 6749. Internet Engineering Task Force (IETF), 2012. URL: https://datatracker.ietf.org/doc/html/rfc6749.

[16] N. Dragoni et al. «Microservices: How To Make Your Application Scale». En: *2017 IEEE International Conference on Cloud Engineering Workshops (IC2EW)*. 2017, págs. 1-6. DOI: 10.1109/IC2EW.2017.8005506.

[17] N. Chawla y R. Jain. «A Tale of Three Videoconferencing Applications: Zoom, Webex and Meet». En: *International Journal of Computer Applications* 183.43 (2021), págs. 26-32. DOI: 10.5120/ijca2021921544.

[18] B. L. Moorhouse y L. Kohnke. «Exploring the influence of Zoom on EFL learners' interaction in virtual classrooms». En: *Computer Assisted Language Learning* 34.3 (2021), págs. 155-172. DOI: 10.1080/09588221.2021.1903052.

[19] N. Naveh, Y. Ravid y A. Zilberman. «Accessibility Comparison of Leading Videoconferencing Platforms for Students with Disabilities». En: *Universal Access in the Information Society* 20.4 (2021), págs. 789-803. DOI: 10.1007/s10209-021-00782-9.

[20] A. Gómez y M. C. Torres. «Adopting Microsoft Teams for Teaching and Learning: A Case Study at a South African University». En: *Education and Information Technologies* 27 (2022), págs. 2201-2225. DOI: 10.1007/s10639-021-10738-5.

[21] J. Y. P. Castillo. «Diseño de arquitectura de software en AWS para Italcol». Tesis de mtría. Pontificia Universidad Javeriana Cali, 2024.

[22] R. C. Delgado. «Creación automática de pipelines para CI/CD usando técnicas de reutilización y variabilidad». Tesis de mtría. Pontificia Universidad Javeriana Cali, 2023.

[23] I. Sommerville. *Software Engineering*. 10th. Harlow, UK: Pearson, 2020.

[24] K. E. Wiegers y J. Beatty. *Software Requirements*. 3rd. Redmond, WA: Microsoft Press, 2014.

[25] J. W. Creswell y V. L. P. Clark. *Designing and Conducting Mixed Methods Research*. 3rd. Thousand Oaks, CA: SAGE, 2018.

[26] *Systems and software engineering — Life cycle processes — Requirements engineering*. Sección 6.4.3. 2018.

[27] B. Kitchenham y S. L. Pfleeger. «Principles of Survey Research Part 1: Turning Lemons into Lemonade». En: *ACM SIGSOFT Software Engineering Notes* 27.3 (2002), págs. 16-18.

[28] K. B. Hass. *The PMI Guide to Business Analysis*. Project Management Institute, 2014.

[29] C. Wohlin. *Experimentation in Software Engineering*. Springer, 2012.

[30] D. A. Dillman, J. D. Smyth y L. M. Christian. *Internet, Phone, Mail, and Mixed-Mode Surveys: The Tailored Design Method*. 4.ª ed. Wiley, 2014.

[31] J. A. Krosnick y S. Presser. «Question and Questionnaire Design». En: *Handbook of Survey Research*. Ed. por P. M.
bibinitperiod J. D. Wright. Emerald, 2010, págs. 263-314.

[32] S. K. Alok Joshi y S. Chandel. «Likert Scale: Explored and Explained». En: *British Journal of Applied Science & Technology* 7.4 (2015), págs. 396-403.

[33] K. Pohl. *Requirements Engineering: Fundamentals, Principles, and Techniques*. Berlin, Germany: Springer, 2010.

[34] ISO/IEC. *Systems and software engineering—Systems and software Quality Requirements and Evaluation (SQuaRE)—System and software quality models (ISO/IEC 25010:2011)*. International Organization for Standardization. 2011.

[35] J. Carifio y R. J. Perla. «Ten Common Misunderstandings, Misconceptions, Persistent Myths and Urban Legends about Likert Scales and Likert Response Formats and Their Antidotes». En: *Journal of Social Sciences* 3.3 (2007), págs. 106-116.

[36] J. Sauro y J. R. Lewis. *Quantifying the User Experience*. Morgan Kaufmann, 2012.

[37] A. Field. *Discovering Statistics Using IBM SPSS Statistics*. 5.ª ed. SAGE, 2018.

[38] J. Corbin y A. Strauss. *Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory*. 4.ª ed. SAGE, 2015.

[39] R. F. DeVellis. *Scale Development: Theory and Applications*. 4.ª ed. SAGE, 2016.

[40] J. W. Creswell. *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches*. 4.ª ed. SAGE, 2014, pág. 127.

[41] I. Etikan, S. A. Musa y R. S. Alkassim. «Comparison of Convenience Sampling and Purposive Sampling». En: *American Journal of Theoretical and Applied Statistics* 5.1 (2016), págs. 1-4.

[42] B. Kitchenham, D. Budgen y P. Brereton. «The Value of Small Samples in Software Engineering Research». En: *Proceedings of the 14th International Conference on Evaluation and Assessment in Software Engineering*. British Computer Society, 2008, págs. 101-106.

[43] G. Eysenbach. «Improving the quality of Web surveys: The Checklist for Reporting Results of Internet E-Surveys (CHERRIES)». En: *Journal of Medical Internet Research* 6.3 (2004), e34. DOI: 10.2196/jmir.6.3.e34.

[44] P. Hoonakker y P. Carayon. «Questionnaire Survey Non-response: A Comparison of Postal Mail and Internet Surveys». En: *Proceedings of the 17th International Ergonomics Association World Congress*. 2009, págs. 1-10.

[45] C. de Colombia. *Ley 1581 de 2012: Protección de Datos Personales*. Diario Oficial No. 48.587, 18 oct. 2012. 2012.

[46] M. D. Wilkinson, M. Dumontier, I. J. Aalbersberg et al. «The FAIR Guiding Principles for Scientific Data Management and Stewardship». En: *Scientific Data* 3 (2016), pág. 160018. DOI: 10.1038/sdata.2016.18.

[47] Y. Perez-Riverol, L. Gatto, H. Hermjakob et al. «Ten Simple Rules for Taking Advantage of Git and GitHub». En: *PLOS Computational Biology* 16.7 (2020), e1008316. DOI: 10.1371/journal.pcbi.1008316.

[48] R. C. Jiménez, M. Kuzak, M. Alhamdoosh et al. «The Carpentries: Teaching Tech Skills to Facilitate Research». En: *The Journal of Open Source Education* 2.11 (2019), pág. 47. DOI: 10.21105/jose.00047.

[49] J. D. Blischak, P. Carbonetto y M. Stephens. «Workflow for Reproducible Data Analysis Using Snakemake and Jupyter Notebooks». En: *F1000Research* 7 (2019), pág. 1955. DOI: 10.12688/f1000research.17405.2.

[50] A. Field. *Discovering Statistics Using IBM SPSS Statistics*. 4th. Sage, 2013.

[51] A. Cavoukian. *Privacy by Design: The 7 Foundational Principles*. Information y Privacy Commissioner of Ontario, 2012.

[52] A. Alzahrani y K. Alshammari. «Security Requirements for Cloud-Based Learning Platforms: A Systematic Review». En: *International Journal of Information Security Science* 10.2 (2021), págs. 289-306.

[53] E. García-Morales y M. Hernández. «Elastic Room Orchestration for Large-Scale Synchronous Classes in the Cloud». En: *IEEE Transactions on Learning Technologies* 16.2 (2023), págs. 271-283.

[54] M. L. McHugh. «The Chi-Square Test of Independence». En: *Biochemia Medica* 23.2 (2013), págs. 143-149. DOI: 10.11613/BM.2013.018.

[55] V. Stodden, P. Guo y Z. Ma. «Toward Reproducible Computational Research: An Empirical Analysis of Data and Code Policy Adoption by Journals». En: *PLoS ONE* 8.6 (2016), e67111. DOI: 10.1371/journal.pone.0067111.

[56] Y. Perez-Riverol, J. Gruning y et al. «Ten Simple Rules for Reproducible Computational Research». En: *PLOS Computational Biology* 18.7 (2022), e1009942. DOI: 10.1371/journal.pcbi.1009942.

[57] J. Saldaña. *The Coding Manual for Qualitative Researchers*. 4th. London, UK: SAGE, 2021.

[58] K. Charmaz. *Constructing Grounded Theory*. 2.ª ed. SAGE, 2014.

[59] L. S. Nowell et al. «Thematic Analysis: Striving to Meet the Trustworthiness Criteria». En: *International Journal of Qualitative Methods* 16 (2017), págs. 1-13. DOI: 10.1177/1609406917733847.

[60] L. Bass, P. Clements y R. Kazman. *Software Architecture in Practice*. 4th. Boston, MA: Addison–Wesley Professional, 2012. ISBN: 978-0-321-81526-8.

[61] *Systems and Software Engineering—Systems and Software Quality Requirements and Evaluation (SQuaRE)—System and Software Quality Models*. ISO/IEC, 2011.

[62] M. Richards y N. Ford. *Fundamentals of Software Architecture: An Engineering Approach*. Sebastopol, CA: O'Reilly Media, 2020. ISBN: 978-1-492-05153-0.

[63] M. A. Babar y R. Capilla. «Capturing and Using Quality Attributes Knowledge in Software Architecture Evaluation Process». En: *Proceedings of the First International Workshop on Managing Requirements Knowledge (MARK'08)*. IEEE. 2008, págs. 1-8. DOI: 10.1109/MARK.2008.4669754.

[64] F. Gilson, M. Galster y F. Georis. «Extracting Quality Attributes from User Stories for Early Architecture Decision Making». En: *Proceedings of the 2019 IEEE International Conference on Software Architecture Companion (ICSA-C)*. IEEE, 2019, págs. 129-136. DOI: 10.1109/ICSA-C.2019.00031.

[65] I. Lytra et al. «Quality Attributes Use in Architecture Design Decision Methods: Research and Practice». En: *Computing* 102.2 (oct. de 2019), págs. 551-572. DOI: 10.1007/s00607-019-00758-9. URL: https://doi.org/10.1007/s00607-019-00758-9.

[66] R. N. Calheiros et al. «SQME: A Framework for Modeling and Evaluation of Software Architecture Quality Attributes». En: *Software Quality Journal* 26 (2018), págs. 2611-2636. DOI: 10.1007/s11219-018-0684-3.

[67] A. Elahi y S. M. Babamir. «Evaluating Software Architectural Styles Based on Quality Features through Hierarchical Analysis and Fuzzy Integral (FAHP)». En: *2015 7th International Conference on Information and Knowledge Technology (IKT)*. IEEE. 2015, págs. 1-7. DOI: 10.1109/IKT.2015.7485970.

[68] S. Moaven y J. Habibi. «A Fuzzy-AHP-Based Approach to Select Software Architecture Based on Quality Attributes (FASSA)». En: *Knowledge and Information Systems* 62.3 (2020), págs. 4569-4597. DOI: 10.1007/s10115-020-01496-7.

[69] F. Ahmad, M. Naeem e I. Ilahi. «Fuzzy MoSCoW: A Fuzzy-Based MoSCoW Method for the Prioritization of Software Requirements». En: *Journal of Intelligent & Fuzzy Systems* 33 (2017), págs. 1235-1243. DOI: 10.3233/JIFS-169289.

[70] S. Brown. *The C4 Model for Visualising Software Architecture*. urlhttps://c4model.com/. InfoQ, 2018.

[71] F. Mårtensson. «Software Architecture Quality Evaluation: Approaches in an Industrial Context». Licentiate Thesis. Karlskrona, Sweden: Blekinge Institute of Technology, 2006. ISBN: 91-7295-082-X.

[72] Microsoft Azure. *¿Qué es la nube?* https://azure.microsoft.com/es-es/resources/cloud-computing-dictionary/what-is-the-cloud/. 2023.

[73] Amazon Web Services. *Precios de AWS*. https://aws.amazon.com/es/pricing/. 2023.

[74] A. W. Services. *AWS Global Infrastructure*. Consultado 22 jul 2025. 2025. URL: https://aws.amazon.com/es/about-aws/global-infrastructure/regions_az/.

[75] Microsoft Azure. *Azure frente a AWS*. https://azure.microsoft.com/es-es/pricing/azure-vs-aws. 2023.

[76] Amazon Web Services. *Beneficios del escalado autom00e1tico para la arquitectura de aplicaciones - Amazon EC2 Auto Scaling*. https://docs.aws.amazon.com/es_es/autoscaling/ec2/userguide/auto-scaling-benefits.html. 2023.

[77] Microsoft Azure. *Introducci00f3n a la escalabilidad autom00e1tica en Azure*. https://learn.microsoft.com/es-es/azure/azure-monitor/autoscale/autoscale-overview. 2025.

[78] Amazon Web Services. *Measuring Global Network Latency with AWS CloudFront*. https://docs.aws.amazon.com/pdfs/whitepapers/latest/secure-content-delivery-amazon-cloudfront/secure-content-delivery-amazon-cloudfront.pdf. Datos de latencia de referencia citados en § Metodología de comparación. 2024.

[79] Microsoft Azure. *Azure Latency Test Performance Guidance*. https://azure.microsoft.com/en-us/blog/advancing-application-reliability-with-performance-testing-in-azure/. Datos de latencia de referencia citados en Metodología de comparación. 2024.

[80] Amazon Web Services. *AWS Service Level Agreements*. https://aws.amazon.com/legal/service-level-agreements/. Consultado el 21 jul 2025. 2025.

[81] Microsoft Azure. *Azure Service Level Agreements*. https://azure.microsoft.com/en-us/support/legal/sla/. Consultado el 21 jul 2025. 2025.

[82] Amazon Web Services. *AWS Pricing Calculator*. https://calculator.aws/#/. Estimaciones de coste usadas en § Metodología de comparación. 2025.

[83] Microsoft Azure. *Azure Pricing Calculator*. https://azure.microsoft.com/en-us/pricing/calculator/. Estimaciones de coste usadas en § Metodología de comparación. 2025.

[84] *Information technology — Security techniques — Information security management systems — Requirements*. ISO/IEC, 2022.

[85] Amazon Web Services. *What is Cloud Computing?* https://aws.amazon.com/what-is-cloud-computing/. Consultado el 21 jul 2025. 2025.

[86] Microsoft Azure. *What is Cloud Computing?* https://azure.microsoft.com/en-us/resources/cloud-computing-dictionary/what-is-cloud-computing. Consultado el 21 jul 2025. 2025.

[87] P. Mell y T. Grance. *The NIST Definition of Cloud Computing*. Inf. téc. Special Publication 800-145. National Institute of Standards y Technology, 2011.

[88] Amazon Web Services. *What is Infrastructure as a Service (IaaS)?* https://aws.amazon.com/what-is/iaas/. Consultado el 22 jul 2025. 2025.

[89] Amazon Web Services. *What is Platform as a Service (PaaS)?* https://aws.amazon.com/what-is/ipaas/. Consultado el 22 jul 2025. 2025.

[90] Amazon Web Services. *What is Software as a Service (SaaS)?* https://aws.amazon.com/what-is/saas/. Consultado el 22 jul 2025. 2025.

[91] Microsoft Azure. *IaaS on Azure—Overview*. https://azure.microsoft.com/es-es/resources/cloud-computing-dictionary/what-is-iaas. Consultado el 22 jul 2025. 2025.

[92] Microsoft Azure. *PaaS on Azure—Overview*. https://azure.microsoft.com/es-es/resources/cloud-computing-dictionary/what-is-paas. Consultado el 22 jul 2025. 2025.

[93] Microsoft Azure. *SaaS on Azure—Overview*. https://azure.microsoft.com/es-es/resources/cloud-computing-dictionary/what-is-saas. Consultado el 22 jul 2025. 2025.

[94] Microsoft Learn. *Shared responsibility in the cloud*. https://learn.microsoft.com/en-us/azure/security/fundamentals/shared-responsibility. Consultado el 22 jul 2025. 2025.

[95] Amazon Web Services. *Tipos de computación en la nube: IaaS, PaaS, Serverless y SaaS*. https://aws.amazon.com/es/types-of-cloud-computing/. Consultado el 22 jul 2025. 2025.

[96] Amazon Web Services. *What is AWS?* https://aws.amazon.com/what-is-aws/. Consultado el 22 jul 2025. 2025.

[97] Amazon Web Services. *AWS Regions and Availability Zones*. https://aws.amazon.com/about-aws/global-infrastructure/regions_az/. Consultado el 22 jul 2025. 2025.

[98] Amazon Web Services. *AWS Well-Architected Framework – Reliability Pillar*. https://docs.aws.amazon.com/wellarchitected/latest/reliability-pillar/. Whitepaper, versión 2023-09. 2023.

[99] Amazon Web Services. *Amazon EC2 Service Level Agreement*. https://aws.amazon.com/compute/sla/. Consultado el 22 jul 2025. 2024.

[100] Amazon Web Services. *How Amazon EC2 Auto Scaling works*. https://docs.aws.amazon.com/autoscaling/ec2/userguide/how-as-works.html. 2025.

[101] Amazon Web Services. *Monitoring the health of EC2 instances in an Auto Scaling group*. https://docs.aws.amazon.com/autoscaling/ec2/userguide/as-instance-health.html. 2025.

[102] Amazon Web Services. *Predictive scaling for Amazon EC2 Auto Scaling*. https://docs.aws.amazon.com/autoscaling/ec2/userguide/ec2-auto-scaling-predictive-scaling.html. 2025.

[103] Amazon Web Services. *Dynamic scaling for Amazon EC2 Auto Scaling*. https://docs.aws.amazon.com/autoscaling/ec2/userguide/ec2-auto-scaling-dynamic-scaling.html. 2025.

[104] Amazon Web Services. *Target tracking scaling policies for Amazon EC2 Auto Scaling*. https://docs.aws.amazon.com/autoscaling/ec2/userguide/as-scaling-target-tracking.html. 2025.

[105] Amazon Web Services. *aws autoscaling put-scaling-policy*. https://awscli.amazonaws.com/v2/documentation/api/latest/reference/autoscaling/put-scaling-policy.html. 2025.

[106] Amazon Web Services. *Amazon EC2 Auto Scaling — FAQs*. https://aws.amazon.com/autoscaling/faqs/. 2025.

[107] A. W. Services. *Savings Plans User Guide*. https://docs.aws.amazon.com/savingsplans/latest/userguide/. Consultado 21 jul 2025. 2025.

[108] A. W. Services. *Reserved Instance Pricing*. https://aws.amazon.com/ec2/pricing/reserved-instances/. Consultado 21 jul 2025. 2025.

[109] A. W. Services. *Amazon EC2 Spot Instances Pricing*. https://aws.amazon.com/ec2/spot/pricing/. Consultado 21 jul 2025. 2025.

[110] A. W. Services. *AWS Cost Explorer User Guide*. https://docs.aws.amazon.com/cost-management/latest/userguide/ce-ug.html. Consultado 21 jul 2025. 2025.

[111] Canalys. «Global cloud services market growth slows to 16 % in Q2 2023». En: *Canalys Newsroom* (2023). AWS lidera con el 30 % del gasto mundial en IaaS/PaaS.

[112] Microsoft. *What is Azure?* https://azure.microsoft.com/en-us/resources/cloud-computing-dictionary/what-is-azure/. Consultado 23 jul 2025. 2025.

[113] M. Docs. *Azure Policy documentation*. https://learn.microsoft.com/azure/governance/policy/overview. Consultado 23 jul 2025. 2025.

[114] M. Docs. *SLA for Virtual Machines*. https://learn.microsoft.com/es-es/azure/virtual-machines/overview. Consultado 23 jul 2025. 2025.

[115] Microsoft. *Azure Expert MSP program*. https://azure.microsoft.com/es-es/partners. Consultado 23 jul 2025. 2024.

[116] Microsoft Azure. *Explore Azure global infrastructure*. Consultado el 23 de julio de 2025. 2025. URL: https://azure.microsoft.com/en-us/explore/global-infrastructure (visitado 23-07-2025).

[117] Microsoft Azure. *Service Level Agreements for Virtual Machines*. Consultado el 23 de julio de 2025. 2025. URL: https://azure.microsoft.com/support/legal/sla/virtual-machines (visitado 23-07-2025).

[118] Microsoft Azure. *Azure reliability: Region pairs*. Consultado el 23 de julio de 2025. 2025. URL: https://learn.microsoft.com/en-us/azure/architecture/aws-professional/regions-zones (visitado 23-07-2025).

[119] Microsoft Azure. *Autoscale overview in Azure Monitor*. Consultado el 23 de julio de 2025. 2025. URL: https://learn.microsoft.com/en-us/azure/azure-monitor/autoscale/autoscale-overview (visitado 23-07-2025).

[120] Microsoft Azure. *Azure Virtual Machine Scale Sets overview*. Consultado el 23 de julio de 2025. 2025. URL: https://learn.microsoft.com/en-us/azure/virtual-machine-scale-sets/overview (visitado 23-07-2025).

[121] Microsoft Azure. *Best practices for autoscale in Azure Monitor*. Consultado el 23 de julio de 2025. 2025. URL: https://learn.microsoft.com/en-us/azure/azure-monitor/autoscale/autoscale-best-practices (visitado 23-07-2025).

[122] Microsoft Azure. *Pricing—Virtual Machine Scale Sets*. Consultado el 23 de julio de 2025. 2025. URL: https://azure.microsoft.com/en-us/pricing/details/virtual-machine-scale-sets (visitado 23-07-2025).

[123] Microsoft Azure. *Azure pricing calculator*. Consultado el 23 de julio de 2025. 2025. URL: https://azure.microsoft.com/en-us/pricing (visitado 23-07-2025).

[124] Microsoft Azure. *Azure Savings Plan for compute*. Consultado el 23 de julio de 2025. 2025. URL: https://learn.microsoft.com/en-us/azure/cost-management-billing/savings-plan/savings-plan-compute-overview (visitado 23-07-2025).

[125] Microsoft Azure. *Reserved Virtual Machine Instances*. Consultado el 23 de julio de 2025. 2025. URL: https://azure.microsoft.com/en-us/pricing/reserved-vm-instances (visitado 23-07-2025).

[126] Microsoft Azure. *Spot Virtual Machines—Eviction policy and pricing*. Consultado el 23 de julio de 2025. 2025. URL: https://azure.microsoft.com/en-us/pricing/spot (visitado 23-07-2025).

[127] Microsoft Azure. *Azure Hybrid Benefit*. Consultado el 23 de julio de 2025. 2025. URL: https://azure.microsoft.com/en-us/pricing/hybrid-use-benefit (visitado 23-07-2025).

[128] Microsoft Azure. *Cost optimization in the Azure Well-Architected Framework*. Consultado el 23 de julio de 2025. 2025. URL: https://learn.microsoft.com/en-us/azure/architecture/framework/cost/overview (visitado 23-07-2025).

[129] M. Azure. *Azure Global Infrastructure*. Consultado 22 jul 2025. 2025. URL: https://azure.microsoft.com/explore/global-infrastructure.

[130] *Azure regions list*. Microsoft Learn, actualizado 2025-05-22. URL: https://learn.microsoft.com/en-us/azure/reliability/regions-list.

[131] Y. Khalidi. *Microsoft partners with the industry to unlock new 5G scenarios with Azure Edge Zones*. Microsoft Azure Blog, 31 Mar 2020. URL: https://azure.microsoft.com/en-us/blog/microsoft-partners-with-the-industry-to-unlock-new-5g-scenarios-with-azure-edge-zones/.

[132] Amazon Web Services. *AWS Elastic Disaster Recovery – Scalable, Cost-Effective Application Recovery*. Consultado el 22 jul 2025. 2025. URL: https://aws.amazon.com/disaster-recovery/.

[133] *About Azure Site Recovery (Service Overview)*. Microsoft Learn, actualizado 2025-04-01. URL: https://learn.microsoft.com/en-us/azure/site-recovery/site-recovery-overview.

[134] N. Tiwari. *Business Continuity and Disaster Recovery for on-premises workloads in Microsoft Azure Cloud*. Microsoft Tech Community Blog, 12 Mar 2024. URL: https://techcommunity.microsoft.com/t5/azure-infrastructure-blog/business-continuity-and-disaster-recovery-for-on-premises-workloads/ba-p/4083157.

[135] Amazon Web Services. *Amazon EC2 Instances*. Consultado 24 jul 2025. 2025. URL: https://aws.amazon.com/ec2/ (visitado 24-07-2025).

[136] Microsoft. *Azure Virtual Machines documentation*. Consultado 24 jul 2025. 2025. URL: https://learn.microsoft.com/azure/virtual-machines/ (visitado 24-07-2025).

[137] Amazon Web Services. *AWS Lambda Developer Guide*. Consultado 24 jul 2025. 2025. URL: https://docs.aws.amazon.com/lambda/ (visitado 24-07-2025).

[138] Microsoft. *Azure Functions—Documentation*. Consultado 24 jul 2025. 2025. URL: https://learn.microsoft.com/azure/azure-functions/ (visitado 24-07-2025).

[139] Amazon Web Services. *Amazon EKS Documentation*. Consultado 24 jul 2025. 2025. URL: https://docs.aws.amazon.com/eks/ (visitado 24-07-2025).

[140] Microsoft. *Azure Kubernetes Service (AKS) documentation*. Consultado 24 jul 2025. 2025. URL: https://learn.microsoft.com/azure/aks/ (visitado 24-07-2025).

[141] Amazon Web Services. *Amazon Simple Storage Service (S3)*. Consultado 24 jul 2025. 2025. URL: https://aws.amazon.com/s3/ (visitado 24-07-2025).

[142] Microsoft. *Azure Blob Storage documentation*. Consultado 24 jul 2025. 2025. URL: https://learn.microsoft.com/azure/storage/blobs/ (visitado 24-07-2025).

[143] Amazon Web Services. *Amazon Relational Database Service (RDS)*. Consultado 24 jul 2025. 2025. URL: https://aws.amazon.com/rds/ (visitado 24-07-2025).

[144] Microsoft. *Azure SQL Database documentation*. Consultado 24 jul 2025. 2025. URL: https://learn.microsoft.com/azure/azure-sql/database/ (visitado 24-07-2025).

[145] Amazon Web Services. *Amazon CloudWatch User Guide*. Consultado 24 jul 2025. 2025. URL: https://docs.aws.amazon.com/cloudwatch/ (visitado 24-07-2025).

[146] Microsoft. *Azure Monitor documentation*. Consultado 24 jul 2025. 2025. URL: https://learn.microsoft.com/azure/azure-monitor/ (visitado 24-07-2025).

[147] Amazon Web Services. *Amazon Route 53 Features*. Consultado 24 jul 2025. 2025. URL: https://aws.amazon.com/route53/features/ (visitado 24-07-2025).

[148] Microsoft. *Azure DNS documentation*. Consultado 24 jul 2025. 2025. URL: https://learn.microsoft.com/azure/dns/ (visitado 24-07-2025).

[149] Amazon Web Services. *AWS Identity and Access Management (IAM)*. Consultado 24 jul 2025. 2025. URL: https://aws.amazon.com/iam/ (visitado 24-07-2025).

[150] Microsoft. *Microsoft Entra ID (Azure AD) documentation*. Consultado 24 jul 2025. 2025. URL: https://learn.microsoft.com/azure/active-directory/ (visitado 24-07-2025).

[151] Amazon Web Services. *Amazon CloudFront—Global Edge Network*. Consultado 24 jul 2025. 2025. URL: https://aws.amazon.com/cloudfront/ (visitado 24-07-2025).

[152] Microsoft. *Azure Front Door Documentation*. Consultado 24 jul 2025. 2025. URL: https://learn.microsoft.com/azure/frontdoor/ (visitado 24-07-2025).

[153] Amazon Web Services. *Amazon API Gateway Developer Guide*. Consultado 24 jul 2025. 2025. URL: https://docs.aws.amazon.com/apigateway/ (visitado 24-07-2025).

[154] Microsoft. *Azure API Management Documentation*. Consultado 24 jul 2025. 2025. URL: https://learn.microsoft.com/azure/api-management/ (visitado 24-07-2025).

[155] Amazon Web Services. *Amazon Simple Queue Service (SQS)*. Consultado 24 jul 2025. 2025. URL: https://aws.amazon.com/sqs/ (visitado 24-07-2025).

[156] Amazon Web Services. *Amazon EventBridge Features*. Consultado 24 jul 2025. 2025. URL: https://aws.amazon.com/eventbridge/ (visitado 24-07-2025).

[157] Microsoft. *Azure Service Bus Documentation*. Consultado 24 jul 2025. 2025. URL: https://learn.microsoft.com/azure/service-bus-messaging/ (visitado 24-07-2025).

[158] Microsoft. *Azure Event Grid Documentation*. Consultado 24 jul 2025. 2025. URL: https://learn.microsoft.com/azure/event-grid/ (visitado 24-07-2025).

[159] Amazon Web Services. *Amazon ElastiCache – Redis & Memcached*. Consultado 24 jul 2025. 2025. URL: https://aws.amazon.com/elasticache/ (visitado 24-07-2025).

[160] Microsoft. *Azure Cache for Redis Documentation*. Consultado 24 jul 2025. 2025. URL: https://learn.microsoft.com/azure/azure-cache-for-redis/ (visitado 24-07-2025).

[161] Amazon Web Services. *Amazon DynamoDB – NoSQL Database*. Consultado 24 jul 2025. 2025. URL: https://aws.amazon.com/dynamodb/ (visitado 24-07-2025).

[162] Microsoft. *Azure Cosmos DB Documentation*. Consultado 24 jul 2025. 2025. URL: https://learn.microsoft.com/azure/cosmos-db/ (visitado 24-07-2025).

[163] Amazon Web Services. *AWS Key Management Service (KMS)*. Consultado 24 jul 2025. 2025. URL: https://aws.amazon.com/kms/ (visitado 24-07-2025).

[164] Microsoft. *Azure Key Vault Documentation*. Consultado 24 jul 2025. 2025. URL: https://learn.microsoft.com/azure/key-vault/ (visitado 24-07-2025).

[165] A. W. Services. *AWS Local Zones*. Consultado 22 jul 2025. 2025. URL: https://aws.amazon.com/about-aws/global-infrastructure/localzones/.

[166] A. W. Services. *AWS Compliance Programs*. Consultado 24 jul 2025. 2025. URL: https://aws.amazon.com/compliance/programs/.

[167] M. Azure. *Azure Compliance Documentation*. Consultado 24 jul 2025. 2025. URL: https://learn.microsoft.com/azure/compliance/.

[168] A. W. Services. *AWS Nitro Enclaves*. Consultado 24 jul 2025. 2024. URL: https://aws.amazon.com/es/ec2/nitro/nitro-enclaves/.

[169] M. Azure. *Azure Confidential Computing*. Consultado 24 jul 2025. 2024. URL: https://azure.microsoft.com/solutions/confidential-compute/.

[170] A. W. Services. *Amazon GuardDuty*. Consultado 24 jul 2025. 2025. URL: https://aws.amazon.com/guardduty/.

[171] M. Azure. *Microsoft Defender for Cloud — Security Overview*. Consultado 24 jul 2025. 2025. URL: https://learn.microsoft.com/azure/defender-for-cloud/.

[172] A. W. Services. *AWS Shared Responsibility Model*. Consultado 24 jul 2025. 2025. URL: https://aws.amazon.com/compliance/shared-responsibility-model/.

[173] A. W. Services. *Elastic Fabric Adapter Performance*. Consultado 24 jul 2025. 2025. URL: https://aws.amazon.com/hpc/efa/.

[174] M. Azure. *HBv4 and HX-series for HPC*. Consultado 24 jul 2025. 2025. URL: https://news.microsoft.com/es-xl/las-maquinas-virtuales-azure-hbv4-y-hx-series-para-hpc-ya-estan-disponibles-de-forma-general/.

[175] A. W. Services. *Amazon EBS gp3 Volumes*. Consultado 24 jul 2025. 2025. URL: https://docs.aws.amazon.com/es_es/ebs/latest/userguide/ebs-volume-types.html.

[176] M. Azure. *Premium SSD v2*. Consultado 24 jul 2025. 2025. URL: https://learn.microsoft.com/azure/virtual-machines/disks-types#premium-ssd-v2.

[177] A. W. Services. *Amazon Aurora Global Database*. Consultado 24 jul 2025. 2025. URL: https://aws.amazon.com/rds/aurora/global-database/.

[178] M. Azure. *Azure Cosmos DB SLA*. Consultado 24 jul 2025. 2025. URL: https://azure.microsoft.com/support/legal/sla/cosmos-db/.

[179] A. W. Services. *AWS Compute SLA*. Consultado 24 jul 2025. 2025. URL: https://aws.amazon.com/compute/sla/.

[180] M. Azure. *Virtual Machines SLA*. Consultado 24 jul 2025. 2025. URL: https://azure.microsoft.com/support/legal/sla/virtual-machines/.

[181] A. W. Services. *S3 Multi-Region Access Points*. Consultado 24 jul 2025. 2025. URL: https://aws.amazon.com/es/s3/features/multi-region-access-points/.

[182] M. Azure. *Azure Storage SLA*. Consultado 24 jul 2025. 2025. URL: https://azure.microsoft.com/support/legal/sla/storage/.

[183] A. W. Services. *AWS Global Accelerator Performance*. Consultado 24 jul 2025. 2025. URL: https://aws.amazon.com/global-accelerator/.

[184] M. Azure. *Azure Front Door Performance*. Consultado 24 jul 2025. 2025. URL: https://learn.microsoft.com/en-us/azure/frontdoor/front-door-overview.

[185] A. W. Services. *AWS Pricing Calculator*. Consultado 24 jul 2025. 2025. URL: https://calculator.aws.

[186] Microsoft. *Azure Pricing Calculator*. Consultado 24 jul 2025. 2025. URL: https://azure.microsoft.com/pricing/calculator.

[187] A. W. Services. *Savings Plans*. Consultado 24 jul 2025. 2025. URL: https://aws.amazon.com/savingsplans.

[188] Microsoft. *Azure Reservations*. Consultado 24 jul 2025. 2025. URL: https://learn.microsoft.com/azure/cost-management-billing/reservations.

[189] A. W. Services. *Amazon S3 Pricing*. Consultado 24 jul 2025. 2025. URL: https://aws.amazon.com/s3/pricing.

[190] Microsoft. *Azure Blob Storage Pricing*. Consultado 24 jul 2025. 2025. URL: https://azure.microsoft.com/pricing/details/storage/blobs.

[191] A. W. Services. *AWS Data Transfer Pricing*. Consultado 24 jul 2025. 2025. URL: https://aws.amazon.com/ec2/pricing/on-demand/#Data_Transfer.

[192] Microsoft. *Azure Bandwidth Pricing*. Consultado 24 jul 2025. 2025. URL: https://azure.microsoft.com/pricing/details/bandwidth.

[193] A. W. Services. *AWS Cost Explorer and Budgets*. Consultado 24 jul 2025. 2025. URL: https://aws.amazon.com/aws-cost-management.

[194] Microsoft. *Azure Cost Management + Billing*. Consultado 24 jul 2025. 2025. URL: https://learn.microsoft.com/es-es/azure/cost-management-billing/.

[195] A. W. Services. *AWS Educate*. Consultado 24 jul 2025. 2025. URL: https://aws.amazon.com/education/awseducate.

[196] Microsoft. *Azure for Students and Faculty*. Consultado 24 jul 2025. 2025. URL: https://azure.microsoft.com/free/students.

[197] A. W. Services. *Amazon EC2 Reserved Instances Pricing*. Consultado 24 jul 2025. 2025. URL: https://aws.amazon.com/ec2/pricing/reserved-instances/.

[198] A. W. Services. *AWS Savings Plans*. Consultado 24 jul 2025. 2025. URL: https://aws.amazon.com/savingsplans/.

[199] A. W. Services. *Amazon EC2 Spot Instances*. Consultado 24 jul 2025. 2025. URL: https://aws.amazon.com/ec2/spot/.

[200] M. Azure. *Azure Spot Virtual Machines*. Consultado 24 jul 2025. 2025. URL: https://learn.microsoft.com/azure/virtual-machines/spot-vms.

[201] M. Azure. *Azure Hybrid Benefit*. Consultado 24 jul 2025. 2025. URL: https://azure.microsoft.com/pricing/hybrid-benefit/.

[202] A. W. Services. *Scheduled Scaling for Amazon EC2 Auto Scaling*. Consultado 24 jul 2025. 2025. URL: https://docs.aws.amazon.com/autoscaling/ec2/userguide/schedule_time.html.

[203] M. Azure. *Autoscale in Azure Monitor*. Consultado 24 jul 2025. 2025. URL: https://learn.microsoft.com/azure/azure-monitor/autoscale/autoscale-overview.

[204] A. W. Services. *Instance Scheduler on AWS*. Consultado 24 jul 2025. 2025. URL: https://aws.amazon.com/solutions/implementations/instance-scheduler-on-aws/.

[205] M. Azure. *Start/Stop VMs during off-hours solution*. Consultado 24 jul 2025. 2025. URL: https://learn.microsoft.com/azure/automation/automation-solution-vm-management.

[206] Amazon Web Services. *Amazon EC2 instance types*. Consultado 24 jul 2025. 2025. URL: https://aws.amazon.com/ec2/instance-types/.

[207] Microsoft. *Sizes for virtual machines in Azure*. Consultado 24 jul 2025. 2025. URL: https://learn.microsoft.com/azure/virtual-machines/sizes.

[208] UNAD. *Campus Agreement/EES con Microsoft*. https://sgeneral.unad.edu.co/00_PUBLICACIONES/INV_PUBLICAS/2024/03/02_TERM_REFE_INVI_PUBL_003.pdf. Consultado el 25 de julio de 2025. 2024.

[209] Amazon Web Services. *AWS Local Zones Locations*. https://aws.amazon.com/es/about-aws/global-infrastructure/localzones/locations/. Región Local Zone en Bogotá anunciada. 2025.

[210] Amazon Web Services. *Dise00f1o de arquitecturas para la nube: Pr00e1cticas recomendadas de AWS*. Whitepaper. 2016. URL: https://d1.awsstatic.com/whitepapers/es_ES/AWS_Cloud_Best_Practices.pdf.

[211] Atlassian. *Ventajas y desventajas de los microservicios que debes conocer*. Blog. 2024. URL: https://www.atlassian.com/es/microservices/cloud-computing/advantages-of-microservices.

[212] Amazon Web Services. *Implementaci00f3n de microservicios en AWS*. Inf. téc. AWS Whitepaper, 2020. URL: https://docs.aws.amazon.com/es_es/whitepapers/latest/microservices-on-aws/microservices-on-aws.pdf.

[213] Amazon Web Services. *00bfQu00e9 son los diagramas de arquitectura?* Sitio web. 2020. URL: https://aws.amazon.com/es/what-is/architecture-diagramming/.

[214] National Institute of Standards and Technology. *Guide for Conducting Risk Assessments*. NIST Special Publication 800-30. Consultado el 23 jul 2025. 2012. URL: https://csrc.nist.gov/publications/detail/sp/800-30/rev-1/final.

[215] Amazon Web Services. *AWS Well-Architected Framework – Security Pillar*. https://docs.aws.amazon.com/wellarchitected/latest/security-pillar/welcome.html. Consultado el 21 jul 2025. 2023.

[216] CloudPingTest.com. *AWS Ping Test (Latency)*. https://cloudpingtest.com/aws. Consulta: 22 jul 2025. 2025.