Diseño de una arquitectura para la automatización de ofertas comerciales en la empresa Infraestructura Virtual SAS

Ricardo Daniel Cordoba Zuñiga

Proyecto de grado entregado para obtener el título de
Magister en Ingeniería de Software

Dirigida por
MSc. Mario Julián Mora Cardona

Pontificia Universidad Javeriana Cali
Facultad de Ingeniería y Ciencias
Maestría en Ingeniería de Software
Santiago de Cali
27 de febrero de 2026

Santiago de Cali, 27 de febrero de 2026.

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 para la automatización de ofertas comerciales en la empresa Infraestructura Virtual SAS" realizado por el estudiante de Magister en Ingeniería de Software Ricardo Daniel Cordoba Zuñiga (cod: 8937189), el cual se encuentra terminado y considero que cumple con los requisitos para ser sustentado.
Atentamente,

MSc. Mario Julián Mora Cardona

Santiago de Cali, 27 de febrero de 2026.

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 para la automatización de ofertas comerciales en la empresa Infraestructura Virtual SAS" 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,

Ricardo Daniel Cordoba Zuñiga
Código: 8937189

Ficha Resumen
Trabajo de Grado Maestría en Ingeniería de Software

TíTULO: Completar

1. Énfasis: Ingeniería de Software
2. Área de trabajo: Ingeniería de Software
3. Tipo de proyecto: Aplicado
4. Estudiante: Ricardo Daniel Cordoba
5. Correo electrónico: rcordoba@javerianacali.edu.co
6. Dirección y teléfono: Calle 60 B #1 07-75 Tel: 3222823760
7. Director: Mario Julian Mora
8. Vinculación del director: Planta
9. Correo electrónico del director: mariomora@javerianacali.edu.co
10. Co-Director (Si aplica):
11. Grupo o empresa que lo avala (Si aplica):
12. Otros grupos o empresas:
13. Palabras clave(al menos 5):
14. ODS que aplica al proyecto (Agenda 2030): ODS 8: Trabajo decente y crecimiento económico; ODS 9: Industria, innovación e infraestructura
15. Fecha de inicio:
16. Resumen: Este proyecto presenta el diseño de una arquitectura de software orientada a automatizar la generación de ofertas comerciales en la empresa Infraestructura Virtual SAS. La propuesta surge como respuesta a problemáticas identificadas en el proceso de preventa, tales como tiempos variables de elaboración, dispersión documental y dependencia del conocimiento individual en la preparación de propuestas.

    La solución propone una arquitectura basada en la nube que integra modelos de inteligencia artificial generativa y un enfoque de recuperación y generación aumentada (RAG) para reutilizar información histórica de manera estructurada. El sistema diseñado permite asistir la elaboración de documentos comerciales, estandarizar estructuras de propuesta y facilitar la incorporación de datos previamente registrados, reduciendo la necesidad de reconstrucción manual de información.

El trabajo se centra en el diseño y evaluación arquitectónica de la solución, incorporando mecanismos explícitos para soportar atributos de calidad como seguridad, rendimiento y fiabilidad. La arquitectura fue analizada mediante escenarios de calidad y el método SAAM, y validada parcialmente a través de una prueba de concepto funcional en entorno controlado.

Como resultado, se obtiene una arquitectura documentada y evaluada que integra capacidades generativas bajo un enfoque desacoplado y gobernado, constituyendo una base técnica para futuras implementaciones orientadas a mejorar la eficiencia operativa y la sostenibilidad tecnológica de la organización.

Agradecimientos

En primer lugar, deseo expresar mi más profundo agradecimiento a todas las personas que hicieron posible la realización de este trabajo de tesis. Este logro no habría sido posible sin el apoyo, la comprensión y la inspiración de quienes me acompañaron a lo largo de este proceso académico y personal.

A mi familia, por su amor incondicional, paciencia y confianza. Gracias por ser mi mayor fuente de motivación y fortaleza, y por recordarme siempre la importancia del esfuerzo, la perseverancia y los valores que guían cada paso de mi vida.

A mis amigos y allegados, quienes con sus palabras de ánimo y compañía constante hicieron más llevadero este camino. Su apoyo emocional y su fe en mis capacidades fueron un impulso invaluable en los momentos de mayor desafío.

A mi director de tesis, Mario Julian Mora, por su orientación, compromiso y disposición para guiarme con claridad, rigor y empatía en cada etapa de la investigación. Su experiencia y aportes fueron fundamentales para la construcción de este trabajo.

A los miembros del comité académico y jurado, por dedicar su tiempo y sus valiosas observaciones para fortalecer la calidad y el alcance de esta investigación.

A mis colegas y compañeros de la maestría, quienes compartieron ideas, experiencias y debates enriquecedores que ampliaron mi perspectiva sobre la arquitectura de software y la innovación tecnológica.

A la empresa Infraestructura Virtual SAS, por permitir el desarrollo de este estudio aplicado y por brindar el contexto real que dio sentido a cada parte de este proyecto. Su apertura a la investigación tecnológica y su disposición para colaborar fueron determinantes para la consolidación de esta propuesta.

Finalmente, agradezco a todas las personas y profesionales que, directa o indirectamente, contribuyeron a este logro. Este trabajo representa no solo un esfuerzo académico, sino también un crecimiento personal y profesional que llevaré conmigo a lo largo de mi trayectoria.

Resumen

Este proyecto propuso el diseño de una arquitectura de software que integró inteligencia artificial para automatizar la generación de ofertas comerciales en la empresa Infraestructura Virtual SAS. La iniciativa respondió a los desafíos actuales del área de preventa, como retrasos en las entregas, errores en la estimación de costos y baja reutilización de propuestas anteriores. Se buscó mejorar la eficiencia operativa, reducir errores humanos y aumentar la competitividad mediante una solución basada en la nube y alineada con las mejores prácticas de arquitectura de software. El sistema automatizado facilitó la estructuración de costos, personalización de ofertas y aprovechamiento de información histórica. Se contempló el uso de tecnologías como modelos generativos, arquitectura RAG y servicios AWS. Se esperó como resultado una arquitectura validada que atendiera atributos clave como seguridad, rendimiento y fiabilidad, sirviendo como base para futuras implementaciones tecnológicas en la empresa.

Palabras Clave: Arquitectura de Software, Inteligencia Artificial Generativa, Automatización de Procesos, Recuperación y Generación Aumentada (RAG), Integración Multivendor, Atributos de Calidad, Evaluación Arquitectónica,SAAM, Arquitectura Cloud, Ofertas Comerciales

Abstract

This project proposed the design of a software architecture that integrated artificial intelligence to automate the generation of commercial offers at Infraestructura Virtual SAS. The initiative addressed current pre-sales challenges such as delivery delays, cost estimation errors, and limited reuse of previous proposals. The goal was to enhance operational efficiency, reduce human errors, and increase competitiveness through a cloud-based solution aligned with best practices in software architecture. The automated system supported cost structuring, offer customization, and historical data reuse. The proposal included technologies such as generative models, Retrieval-Augmented Generation (RAG) architecture, and AWS services. The expected outcome was a validated architecture that met key quality attributes—security, performance, and reliability—providing a foundation for future technological implementations in the company.

Keywords: Automation, RAG, Software Architecture, Quality Attributes, C4, Neural Networks, Deep Learning, AWS, Commercial Proposals.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Introducción

La transformación digital ha modificado de manera sustancial la forma en que las organizaciones estructuran sus procesos internos y generan valor a partir de la información. En este contexto, la incorporación de capacidades de inteligencia artificial generativa en entornos empresariales se consolidó como una alternativa viable para optimizar tareas intensivas en conocimiento y documentación. No obstante, su adopción planteó desafíos técnicos y arquitectónicos que trascendieron la simple integración de modelos, requiriendo mecanismos explícitos de control, gobernanza y aseguramiento de atributos de calidad.

El presente trabajo tuvo como propósito diseñar una arquitectura de software orientada a automatizar la generación de ofertas comerciales en la empresa Infraestructura Virtual SAS. El proyecto surgió como respuesta a problemáticas identificadas en el proceso de preventa, tales como tiempos variables de elaboración, reprocesos asociados a la reconstrucción manual de información y dependencia del conocimiento individual para la estructuración de propuestas. Estas condiciones afectaban la eficiencia operativa y limitaban la capacidad de estandarización y reutilización del conocimiento corporativo.

Frente a este escenario, se planteó como objetivo general diseñar una arquitectura de software basada en la nube que integrara inteligencia artificial generativa bajo un enfoque multivendor, garantizando alineación con atributos de calidad como seguridad, rendimiento, fiabilidad y escalabilidad. Para alcanzar este propósito, se definieron objetivos específicos orientados al análisis del proceso actual, la identificación de mecanismos arquitectónicos adecuados, el diseño de un componente central de integración (Generative AI Hub), la implementación de una prueba de concepto y la evaluación de la solución mediante escenarios de calidad y el método SAAM.

Metodológicamente, el trabajo adoptó un enfoque aplicado de diseño y evaluación arquitectónica. En una primera fase se realizó un análisis del proceso de negocio y de las necesidades organizacionales. Posteriormente, se definió la arquitectura propuesta utilizando modelos de descomposición estructural, incluyendo el modelo C4, y se identificaron decisiones de diseño orientadas a soportar atributos de calidad. La validación se llevó a cabo mediante evaluación basada en escenarios, aplicando principios del método SAAM para analizar riesgos y capacidad de evolución. Finalmente, se desarrolló una prueba de concepto funcional en entorno controlado con el fin de verificar la viabilidad técnica de los componentes críticos de la arquitectura.

La relevancia de este trabajo radicó en la necesidad creciente de integrar inteligencia artificial generativa en procesos empresariales de manera estructurada y sostenible. En lugar de abordar la automatización como una simple adopción tecnológica, la propuesta se sustentó en un enfoque arquitectónico que permitió desacoplar proveedores, aplicar políticas transversales y mantener trazabilidad sobre las decisiones automatizadas.

El desarrollo del proyecto permitió trasladar principios consolidados de arquitectura de software al diseño de una solución empresarial basada en inteligencia artificial generativa, evidenciando que estas ca-

pacidades pueden integrarse sin comprometer criterios formales de calidad. Asimismo, la solución diseñada constituye una base técnica que puede orientar futuras implementaciones dentro de la organización, favoreciendo la adopción progresiva de mecanismos de automatización y la reutilización estructurada del conocimiento corporativo.

El documento se estructura de la siguiente manera. En el Capítulo 1 se presenta el contexto del problema y la definición formal de los objetivos. El Capítulo 2 desarrolla el marco teórico relacionado con arquitectura de software, atributos de calidad e inteligencia artificial generativa. El Capítulo 3 describe el análisis del proceso actual de generación de ofertas. El Capítulo 4 expone el diseño de la arquitectura propuesta, incluyendo el Generative AI Hub y la estrategia de enrutamiento dinámico. El Capítulo 5 presenta la evaluación arquitectónica mediante escenarios y el método SAAM. El Capítulo 6 describe la prueba de concepto y los resultados obtenidos. Finalmente, el Capítulo 7 consolida las conclusiones, trabajos futuros y lecciones aprendidas.

1.2. Definición del problema

1.2.1. Planteamiento del problema

En el sector de tecnología de la información (TI), la generación de propuestas comerciales representó un proceso crítico dentro del ciclo de ventas, ya que determinaba la capacidad de las empresas para competir, adaptarse a los tiempos de respuesta del mercado y presentar ofertas precisas y sostenibles. Este proceso implicaba múltiples etapas, tales como el análisis de requerimientos, la estimación de costos, la personalización de documentos y la estructuración de soluciones técnicas (Lucidchart, 2025). En la práctica, muchas organizaciones continuaban ejecutando estas actividades de forma manual, lo que generaba ineficiencias, errores humanos, baja reutilización de información previa y escasa trazabilidad.

Infraestructura Virtual SAS enfrentaba estos desafíos de manera recurrente. Su equipo de preventa, conformado por tres (3) ingenieros, gestionaba en promedio doce (12) oportunidades semanales a través del CRM, con un plazo de entrega estimado de cinco días. Sin embargo, únicamente el 25 % de estas propuestas se entregaba dentro del tiempo esperado. Entre las causas identificadas se encontraban la elaboración manual de documentos, la dependencia del conocimiento individual del equipo y la influencia de procesos externos, como las demoras en la recepción de cotizaciones de proveedores, las cuales podían extenderse hasta ocho días hábiles. Si bien estos factores externos no eran completamente controlables, sí se identificó la posibilidad de mitigar su impacto mediante la automatización de componentes reutilizables y la estructuración inteligente de información histórica.

Esta situación afectaba la capacidad de respuesta frente a los clientes y reducía la competitividad en licitaciones y oportunidades comerciales. Además, la ausencia de mecanismos de apoyo automatizado incrementaba el riesgo de inconsistencias en la estimación de costos y recursos. Se observó también una alta similitud entre propuestas previas, lo que evidenciaba un potencial significativo de reutilización. No obstante, la falta de herramientas tecnológicas adecuadas obligaba a reconstruir documentos desde cero en cada oportunidad, aumentando la carga operativa y la probabilidad de errores.

Ante este panorama, se identificó la necesidad de estructurar una solución que no se limitara a la incorporación puntual de herramientas de inteligencia artificial, sino que definiera una arquitectura integral capaz de soportar automatización, reutilización de conocimiento y control operativo. En consecuencia, se planteó el diseño de una arquitectura de software basada en la nube que integrara capacidades de

inteligencia artificial generativa mediante un enfoque de hub desacoplado, permitiendo la interacción con múltiples proveedores a través de APIs estandarizadas.

La propuesta contempló la incorporación de mecanismos de recuperación y generación aumentada (RAG) para aprovechar información histórica de propuestas anteriores, facilitando la contextualización del contenido generado. Asimismo, se definió una estrategia de enrutamiento dinámico para seleccionar el proveedor de IA más adecuado según criterios operativos. Todo ello se diseñó bajo principios de arquitectura cloud y alineado con atributos de calidad tales como seguridad, rendimiento, fiabilidad y escalabilidad, con el propósito de garantizar que la solución pudiera evolucionar de manera sostenible dentro del entorno empresarial.

1.2.2. Formulación del problema

¿Cómo diseñar una arquitectura de software que incluya componentes de inteligencia artificial multivendor para automatizar la generación de ofertas comerciales de proyectos de TI en Infraestructura Virtual SAS?

Sistematización:

- ¿Cómo encontrar puntos críticos de mejora, oportunidades de automatización y reutilización de información de propuestas comerciales pasadas?
- ¿Qué componentes arquitectónicos se deben integrar para maximizar los atributos de calidad seguridad, rendimiento, fiabilidad y escalabilidad en la solución?
- ¿Qué mecanismos técnicos permiten integrar múltiples modelos de inteligencia artificial tipo RAG a través de APIs, y cómo pueden operar como un hub flexible dentro de la arquitectura?
- ¿Cómo validar mediante una prueba de concepto (PoC) la generación automática de propuestas comerciales?
- ¿ Cómo evaluar la arquitectura propuesta en coherencia con los requisitos de calidad, sostenibilidad técnica y aplicabilidad real en el entorno empresarial?

1.3. Objetivos del proyecto

1.3.1. Objetivo General

Diseñar una arquitectura de software que automatice la generación de ofertas comerciales en Infraestructura Virtual SAS, integrando componentes de inteligencia artificial multivendor, con un enfoque flexible, reutilizable, y alineado con principios de calidad y escalabilidad en la nube.

1.3.2. Objetivos específicos

OE1: Analizar detalladamente el proceso actual de generación de propuestas comerciales en la empresa para hallar puntos críticos de mejora, oportunidades de automatización y reutilización de información.

OE2: Definir los elementos de la arquitectura que maximicen los atributos de calidad seguridad, rendimiento, fiabilidad y escalabilidad.

OE3: Diseñar una arquitectura flexible basada en principios de computación en la nube, que permita la integración desacoplada de múltiples mecanismos de IA tipo RAG a través de APIs, operando como un hub de orquestación inteligente.

OE4: Desarrollar una prueba de concepto (PoC) que genere automáticamente propuestas comerciales con todos sus apartados fundamentales alcance técnico, beneficios, condiciones, restricciones y valores económicos; reutilizando propuestas existentes.

OE5: Evaluar la arquitectura propuesta mediante escenarios definidos y el método SAAM, validando su alineación con los requisitos de calidad, sostenibilidad técnica y aplicabilidad real en el entorno empresarial.

1.4. Delimitaciones y alcances

- La arquitectura propuesta fue diseñada para ser implementada en la empresa Infraestructura Virtual SAS y tuvo como objetivo automatizar el proceso de generación de ofertas comerciales en el área de preventa.
- Se realizó el análisis del proceso actual de elaboración de propuestas comerciales, incluyendo sus limitaciones, tiempos de respuesta y puntos críticos.
- Se evaluó la arquitectura utilizando el método SAAM, a partir de escenarios definidos.
- Se desarrolló una prueba de concepto (PoC) que permitió validar parcialmente el comportamiento del sistema, su capacidad para reutilizar información histórica de propuestas comerciales y su impacto en la mejora de los tiempos de preparación y plazos de entrega de las propuestas comerciales.
- No se incluyó la evaluación de los costos asociados a una futura implementación en producción.
- Ya se contaba con la autorización por parte de Infraestructura Virtual SAS para el uso de información interna de carácter no sensible con fines académicos, como flujos de trabajo, documentos de propuestas previas y ejemplos comerciales.
- En caso de haberse requerido información adicional durante el desarrollo o validación del proyecto, esta se gestionó directamente con la compañía bajo criterios de confidencialidad.

1.5. Justificación del trabajo de grado

En la industria de TI, la rapidez y precisión en la generación de ofertas comerciales son determinantes para la competitividad y sostenibilidad empresarial. Infraestructura Virtual SAS actualmente enfrenta múltiples desafíos en la elaboración manual de propuestas comerciales, lo que afecta su capacidad de respuesta y reduce su competitividad en licitaciones y oportunidades de negocio. Si la empresa no implementa una solución automatizada basada en inteligencia artificial, los impactos negativos serán cuantificables y representarán un riesgo significativo para su crecimiento.

Si no se implementa la automatización para la generación de ofertas comerciales la empresa enfrentará los siguientes impactos negativos:

- Retrasos en la entrega de propuestas: Actualmente, solo el 25 % de las propuestas se entregan dentro del plazo estipulado de cinco días, mientras que el 75 % restante sufre retrasos de hasta ocho días debido a la dependencia de procesos manuales. Esta ineficiencia reduce la tasa de éxito en licitaciones, limitando la captación de nuevos proyectos. Este comportamiento es consistente con tendencias documentadas en la industria: según QorusDocs (2026), el 76 % de los equipos de TI y SaaS reportan que sus propuestas tardan más de seis días en completarse, y el 50 % señala no contar con recursos suficientes para manejar el volumen de solicitudes.

- Errores en la estimación de costos y recursos: La falta de precisión en el cálculo de costos genera sobrecostos de hasta un 15 % por proyecto, afectando la rentabilidad. Asimismo, la subestimación de recursos conlleva retrasos en la ejecución, lo que impacta negativamente la relación con los clientes y la reputación de la empresa. Esta problemática está documentada a nivel global: Bloch et al. (2012) encontraron que el 66 % de los proyectos de software empresarial presentan sobrecostos, y que cada año adicional en la duración de un proyecto incrementa el desfase presupuestal en un 15 %.

- Sobrecarga operativa en el equipo de preventa: Actualmente, el equipo de preventa debe gestionar en promedio doce oportunidades de negocio semanales, lo que equivale a más de 50 propuestas al mes. Sin una automatización eficiente, el tiempo invertido en tareas repetitivas y manuales podría representar un incremento del 30 % en la carga de trabajo, reduciendo la capacidad de respuesta y aumentando el riesgo de errores humanos. De acuerdo con QorusDocs (2026), el 42 % de los equipos de TI y SaaS gestionan entre 10 y 24 propuestas proactivas al mes, confirmando que este volumen representa uno de los mayores desafíos operativos del sector.

- Reproceso y falta de reutilización de información histórica: La ausencia de un sistema que permita almacenar y reutilizar datos históricos obligará a los especialistas en preventa a reconstruir propuestas desde cero en cada oportunidad, lo que representa una ineficiencia operativa y una pérdida de recursos valiosos.

- Dependencia de procesos externos como cotizaciones de proveedores: aunque no pueden controlarse directamente, su impacto puede mitigarse si se cuenta con plantillas reutilizables, componentes inteligentes precargados y herramientas de automatización que reduzcan la necesidad de iniciar cada propuesta desde cero.

- Menor capacidad de adaptación y escalabilidad: Con el crecimiento del negocio, la demanda de generación de ofertas también aumentará. Sin una solución automatizada, la empresa se enfrentará a un cuello de botella operativo que limitará su capacidad para escalar sus procesos comerciales de manera eficiente.

  Desde un punto de vista técnico, la no implementación de esta solución significará una brecha tecnológica importante con respecto a las tendencias actuales en arquitecturas en la nube, automatización e inteligencia artificial. La empresa seguirá dependiendo de metodologías manuales obsoletas que restringen su capacidad de innovación y evolución en el mercado.

En conclusión, la diseño de una arquitectura para la automatización de ofertas comerciales con componentes de inteligencia artificial permitirá a Infraestructura Virtual SAS mejorar su eficiencia operativa, mitigar riesgos, reutilizar conocimiento y mantenerse competitiva en un entorno cada vez más dinámico y exigente. No adoptar esta solución representa un riesgo significativo de rezago tecnológico y comercial, lo que podría comprometer la sostenibilidad y el crecimiento de la empresa a mediano y largo plazo.

1.6. Metodología de la investigación

La metodología de este trabajo se desarrolló bajo un enfoque ágil inspirado en Scrum, organizado en ciclos de trabajo de dos semanas (15 días). Este esquema permitió avanzar de manera progresiva, revisando cada etapa antes de continuar con la siguiente y ajustando decisiones conforme el proyecto iba tomando forma.

Cada quince (15) días se realizaron reuniones con el director del proyecto para revisar avances, aclarar dudas y redefinir prioridades cuando fue necesario. Las actividades quedaron registradas en un cronograma que facilitó el seguimiento de los entregables parciales y el control del progreso general.

El trabajo se estructuró en iteraciones que incluyeron planificación, ejecución y revisión de resultados. Esta dinámica favoreció una comunicación constante, permitió incorporar retroalimentación oportuna y mantuvo el proyecto alineado con los objetivos planteados.

Este enfoque facilitó organizar el desarrollo del proyecto de manera coherente, desde el análisis inicial del problema hasta la validación de la solución propuesta, asegurando una construcción progresiva y documentada de cada decisión adoptada.

1.6.1. Fases del proyecto

El desarrollo del proyecto se organizó en cinco fases principales, articuladas de manera iterativa y alineadas con los objetivos definidos:

1.6.1.1. Diagnóstico del proceso actual

- Se recopiló información detallada sobre el proceso de generación de ofertas en Infraestructura Virtual SAS.
- Se analizaron los flujos de trabajo, los tiempos de respuesta, las herramientas utilizadas y las principales limitaciones operativas.
- Se identificaron los puntos críticos del proceso, los cuellos de botella y las oportunidades concretas de mejora mediante automatización.

1.6.1.2. Definición de requisitos y atributos de calidad

- Se levantaron y validaron los requisitos funcionales y no funcionales del sistema.
- Se definieron y priorizaron los atributos de calidad más relevantes para el contexto empresarial.

- Se contrastaron estos requisitos con las necesidades del negocio y las expectativas asociadas a la automatización del proceso.

1.6.1.3. Diseño de la arquitectura

- Se construyeron los diagramas arquitectónicos utilizando el modelo C4, abordando los niveles de contexto, contenedores y componentes.
- Se documentaron las decisiones de diseño más relevantes y su justificación técnica.
- Se diseñó el componente central de integración de IA tipo RAG bajo un enfoque multivendor mediante APIs, como pieza clave de la arquitectura.

1.6.1.4. Evaluación de la arquitectura

- Se aplicó el método SAAM para analizar riesgos, capacidad de evolución y alineación con los atributos de calidad definidos.
- Se formularon escenarios de evaluación y se documentaron los resultados obtenidos.
- Se incorporó retroalimentación técnica especializada y se realizaron los ajustes necesarios en el diseño.

1.6.1.5. Desarrollo de una prueba de concepto (PoC)

- Se implementó un módulo funcional que permitió automatizar parcialmente la generación de ofertas, reutilizando información histórica disponible.
- Se observó su comportamiento frente al proceso manual, considerando tiempos de respuesta, coherencia del contenido y consistencia documental.

Capítulo 2

Marco de referencia

2.1. Marco Teórico

2.1.1. Inteligencia artificial generativa

La Inteligencia Artificial Generativa (GAI) se refiere a la capacidad de los sistemas de IA para generar contenido nuevo y original, incluyendo texto, imágenes, música y otros formatos, a partir del análisis de datos y patrones previamente aprendidos. Esta tecnología, considerada como una de las áreas más dinámicas dentro del campo de la inteligencia artificial, automatiza la producción de contenido mediante el aprendizaje adaptativo y el procesamiento multimodal.

Según Liu (2023), la IA generativa no solo introduce nuevas formas de innovación tecnológica, sino que también transforma profundamente la industria de los medios, facilitando la producción automatizada de noticias, la curaduría de contenido y la personalización de la información según las necesidades del usuario. No obstante, su implementación plantea desafíos éticos y regulatorios, como la generación de noticias falsas, la privacidad de los usuarios y la necesidad de mecanismos de supervisión efectivos. Para aprovechar su potencial de manera responsable, la industria debe adoptar estrategias que equilibren la innovación con la seguridad y la transparencia en la generación de contenido.

Entre las IA generativas más recientes se encuentran GPT-5.4 (OpenAI, 2026), Gemini 3.1 Pro Preview (Google DeepMind, 2026), Claude Opus 4.6 y Claude Sonnet 4.6 (Anthropic, 2026), modelos que lideran el índice de inteligencia de Artificial Analysis a abril de 2026 (Artificial Analysis, 2026).

GPT-5.4, desarrollado por OpenAI y lanzado en marzo de 2026, integra capacidades nativas de uso de computador y una ventana de contexto de un millón de tokens, posicionándose como modelo líder para trabajo profesional complejo (OpenAI, 2026).

Gemini 3.1 Pro Preview, de Google DeepMind, lidera benchmarks de razonamiento como ARC-AGI-2 con un 77.1 %, optimizando tareas de síntesis de información compleja y flujos de trabajo agénticos (Google DeepMind, 2026).

Claude Opus 4.6 y Claude Sonnet 4.6, desarrollados por Anthropic, destacan en razonamiento y codificación, manteniendo el énfasis en seguridad y alineación ética que caracteriza a Claude (Anthropic, 2026).

Estos modelos lideran el índice de inteligencia de Artificial Analysis a abril de 2026, con scores de 57, 57, 53 y 52 respectivamente (Artificial Analysis, 2026).

2.1.2. Redes neuronales y modelos de aprendizaje profundo.

Las redes neuronales artificiales (ANNs) son modelos computacionales inspirados en la estructura del cerebro humano, diseñados para procesar y analizar grandes volúmenes de datos a través de capas de neuronas interconectadas. Estas redes son fundamentales en el campo del aprendizaje profundo (Deep

Learning), una rama del aprendizaje automático (Machine Learning) que utiliza múltiples capas para extraer características y patrones complejos en los datos (Salas et al., 2019)

- Redes Neuronales Convolucionales (CNNs): Especializadas en el procesamiento de datos con estructuras espaciales, como imágenes y secuencias de información.
- Redes Neuronales Recurrentes (RNNs): Diseñadas para analizar datos secuenciales, con aplicaciones en procesamiento de lenguaje natural y series temporales.
- Transformers: Modelos avanzados utilizados en tareas de procesamiento de lenguaje, capaces de manejar grandes volúmenes de texto con alta precisión.

2.1.3. Arquitectura RAG

La Arquitectura de Recuperación y Generación Aumentada (RAG) es un enfoque en inteligencia artificial que combina la recuperación de información con modelos generativos para mejorar la precisión y relevancia de las respuestas en tareas de procesamiento del lenguaje natural. Su funcionamiento se basa en dos componentes:

Componente de Recuperación: Obtiene información relevante desde bases de datos externas o fuentes documentales utilizando métodos como búsqueda semántica o embeddings vectoriales.

Componente de Generación: Usa la información recuperada como contexto adicional para un modelo de lenguaje preentrenado, lo que permite generar respuestas más precisas y contextualizadas. Este enfoque reduce la alucinación del modelo y mejora su capacidad de actualización, ya que complementa su conocimiento con datos externos recientes. La integración de modelos de lenguaje con bases de datos vectoriales en un entorno en la nube mejora la comprensión contextual y la precisión de las respuestas generadas, mitigando los problemas de desactualización y generación errónea de información (Boppana et al., 2024).

2.1.4. Desacoplamiento y orquestación de servicios de IA (hub de IA)

El desacoplamiento y la orquestación de servicios de inteligencia artificial son principios arquitectónicos para construir soluciones escalables, flexibles y sostenibles que integran múltiples modelos o motores de IA.

- El desacoplamiento se refiere a la separación lógica entre el sistema principal (la solución empresarial) y los servicios de IA que utiliza. En lugar de integrar directamente un único proveedor o modelo de IA, el sistema consume los servicios de forma indirecta a través de interfaces estándar (APIs), facilitando la interoperabilidad, el reemplazo dinámico de componentes y la evolución tecnológica a lo largo del tiempo (Richards and Ford, 2020).
- La orquestación, por su parte, implica la capacidad de coordinar múltiples servicios de IA desde un único componente central. Este componente, comúnmente conocido como hub de IA, actúa como un intermediario inteligente que selecciona, enruta, combina y gestiona las interacciones con distintos modelos de lenguaje, motores de inferencia o plataformas de IA generativa. El hub puede aplicar reglas dinámicas para elegir el proveedor más adecuado según el contexto, el tipo de consulta, el costo o la disponibilidad (HatchWorks, 2024).

Este enfoque permite que una arquitectura no dependa de un solo proveedor de inteligencia artificial (vendor lock-in), facilite la integración de nuevos modelos o tecnologías emergentes, y mantenga una estructura modular que soporte tanto pruebas de concepto como entornos de producción a gran escala.

2.1.5. Enrutamiento de modelos de IA (AI model routing)

El routing de modelos de IA (AI model routing) es una estrategia utilizada en arquitecturas multivendor para determinar dinámicamente qué modelo de lenguaje utilizar en función de criterios como el tipo de tarea, la latencia, el costo, la disponibilidad o el rendimiento esperado. Esta técnica permite construir soluciones inteligentes que integran múltiples modelos generativos (LLMs) como GPT, Claude, LLaMA, entre otros, desde un componente centralizado o hub orquestador (Amazon Web Services, 2024).

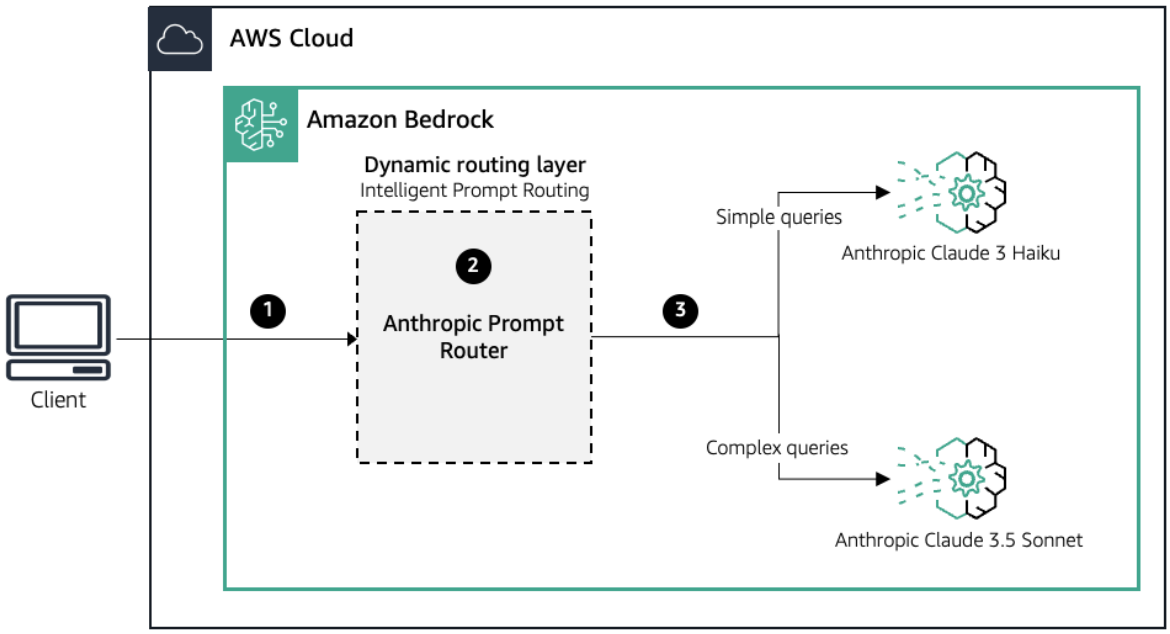


Figura 2.1: Diagrama rutas dinámicas. Fuente: Seifi and Chugh (2025).

La Figura 2.1 ilustra el funcionamiento del enrutamiento dinámico dentro de una arquitectura multillm. El proceso puede comprenderse en tres etapas principales. En la etapa (1), el cliente envía la solicitud al servicio de inferencia, el cual recibe el *prompt* y lo canaliza hacia la capa de enrutamiento. En la etapa (2), un componente especializado —denominado *Prompt Router*— analiza el contenido de la solicitud y evalúa su naturaleza, aplicando reglas predefinidas o modelos clasificadores ligeros para determinar el nivel de complejidad y los recursos necesarios para su procesamiento. Finalmente, en la etapa (3), la consulta es redirigida al modelo más adecuado según la clasificación realizada; por ejemplo, un modelo optimizado para baja latencia y menor costo en el caso de consultas simples, o un modelo de mayor capacidad y precisión para tareas más complejas.

Este enfoque permite optimizar el uso de recursos computacionales, mejorar la eficiencia operativa y

establecer mecanismos de balanceo de carga y tolerancia a fallos. En consecuencia, el enrutamiento dinámico se consolida como un componente clave en arquitecturas generativas modernas, donde la variabilidad en la demanda y el costo de inferencia requieren estrategias inteligentes de asignación de modelos.

2.1.6. Embeddings y representación vectorial

Los modelos fundacionales transforman texto en representaciones numéricas denominadas embeddings, las cuales permiten capturar relaciones semánticas en espacios vectoriales de alta dimensión. Estas representaciones facilitan el cálculo de similitud entre documentos mediante métricas como distancia coseno o producto punto (Jurafsky and Martin, 2023).

En aplicaciones empresariales, los embeddings constituyen la base para mecanismos de recuperación semántica, ya que permiten indexar documentos históricos y compararlos con nuevas consultas en función de su cercanía conceptual (Huyen, 2023). Las guías técnicas actuales para aplicaciones LLM recomiendan el uso de embeddings como componente esencial en arquitecturas de recuperación aumentada (OpenAI, 2023a).

2.1.7. Bases de datos vectoriales y recuperación semántica

Las bases de datos vectoriales permiten almacenar representaciones embebidas de documentos y realizar búsquedas basadas en similitud semántica en lugar de coincidencia exacta de palabras clave. Este enfoque resulta especialmente adecuado para aplicaciones de recuperación aumentada (RAG), donde la precisión contextual es prioritaria (Pinecone, 2023).

Las arquitecturas modernas de búsqueda vectorial utilizan estructuras optimizadas para consultas de vecinos más cercanos, permitiendo escalar a grandes volúmenes de información con tiempos de respuesta controlados (Google Cloud, 2023). Este tipo de almacenamiento difiere de los sistemas relacionales tradicionales, ya que prioriza proximidad vectorial sobre igualdad estructural.

2.1.8. Control de tráfico y limitación de solicitudes

El control de tráfico mediante técnicas de rate limiting permite proteger sistemas distribuidos frente a sobrecargas y picos inesperados de demanda. Estrategias como token bucket o throttling ayudan a mantener estabilidad operativa y prevenir degradación del servicio (Beyer et al., 2022).

Los marcos de arquitectura cloud recomiendan incorporar mecanismos de control de consumo y límites por cliente como parte de las buenas prácticas de confiabilidad y seguridad (Amazon Web Services, 2023).

2.1.9. Observabilidad en sistemas cloud-native

La observabilidad permite comprender el comportamiento interno de un sistema a partir de métricas, registros y trazas distribuidas. En entornos desacoplados, esta práctica resulta esencial para diagnosticar fallos y mantener continuidad operativa (Beyer et al., 2022).

La adopción de estándares abiertos para telemetría facilita la integración de métricas y monitoreo en arquitecturas modernas basadas en servicios distribuidos.

2.1.10. Automatización de procesos empresariales

La automatización de procesos empresariales (BPA, Business Process Automation) es una estrategia que busca optimizar la ejecución de tareas repetitivas dentro de una organización mediante el uso de tecnologías avanzadas. Tradicionalmente, los sistemas de automatización han estado basados en flujos de trabajo rígidos y reglas predefinidas, lo que limita su capacidad de adaptación a entornos dinámicos y a la evolución de los procesos empresariales (Paulose and Neelanath, 2024). Sin embargo, la incorporación de inteligencia artificial generativa ha permitido superar estas limitaciones, transformando la automatización en un sistema flexible que aprende y evoluciona a partir de datos históricos, permitiendo la generación autónoma de nuevas reglas y procesos.

2.1.11. Arquitectura de Sofware

"La arquitectura de software es el conjunto de estructuras de un sistema, compuesto por elementos de software, las relaciones entre ellos y las propiedades de estos elementos y relaciones."(Bass et al., 2022)

2.1.12. Patrones de resiliencia en arquitecturas distribuidas

Los sistemas distribuidos que dependen de servicios externos deben incorporar mecanismos de tolerancia a fallos para evitar la propagación de errores y la degradación del servicio. En arquitecturas modernas, patrones como circuit breaker, retry y fallback se consideran fundamentales para mantener la estabilidad operativa cuando existen dependencias remotas (Newman, 2022).

El patrón Circuit Breaker permite interrumpir llamadas hacia servicios que presentan fallos repetitivos, evitando que una dependencia inestable comprometa el funcionamiento global del sistema (Microsoft Azure Architecture Center, 2023). Asimismo, los lineamientos de arquitectura en la nube enfatizan la necesidad de diseñar para fallos, promoviendo estrategias de resiliencia y recuperación controlada (Amazon Web Services, 2023).

2.1.13. Modelo C4

El modelo C4, creado por (Brown, 2017), es un enfoque estructurado para visualizar la arquitectura de software en cuatro niveles jerárquicos. Se basa en la idea de representar el sistema desde diferentes perspectivas, permitiendo que diferentes audiencias (desde ejecutivos hasta desarrolladores) comprendan cómo está diseñado el software (Vázquez-Ingelmo et al., 2020)

Este modelo se alinea con los principios de (Richards and Ford, 2020), ya que ellos enfatizan:

- La necesidad de diagramas arquitectónicos comprensibles.
- La modularidad en el diseño de sistemas.
- La importancia de representar la arquitectura en múltiples niveles de detalle.

Diagrama de Contexto El Diagrama de Contexto es el nivel más alto del modelo C4 y proporciona una visión general del sistema de software dentro de su entorno,se busca representar los usuarios y los sistemas externos que interactúan con el software (Brown, 2017). Elementos clave:

- Actores (personas o sistemas externos) que interactúan con el sistema.
- Relación entre el sistema y los actores mediante líneas de comunicación.
- Servicios o datos que los sistemas externos proporcionan al software.

  Este nivel es útil para stakeholders no técnicos porque muestra el propósito del sistema sin entrar en detalles técnicos.

Diagrama de Contenedores El Diagrama de Contenedores desglosa el sistema en unidades principales de software llamadas çontenedores". El objetivo es mostrar la estructura de alto nivel del sistema y cómo sus partes principales interactúan (Brown, 2017).

Elementos clave:

- Contenedores (aplicaciones, bases de datos, microservicios, etc.).
- Relación entre contenedores mediante flechas indicando la comunicación.
- Función de cada contenedor en el sistema.

Es útil para desarrolladores y arquitectos, ya que define los principales bloques de construcción de la arquitectura.

Diagrama de Componentes El Diagrama de Componentes desglosa cada contenedor en sus partes internas.El objetivo es mostrar los componentes dentro de cada contenedor y sus interacciones (Brown, 2017).

Elementos clave:

- Componentes dentro de un contenedor (por ejemplo, módulos o servicios internos).
- Relación entre los componentes y su comunicación con otros contenedores.
- Descripción de la función de cada componente.

Ayuda a los desarrolladores a entender la organización interna de una aplicación o servicio.

2.1.14. Software Architecture Analysis Method (SAAM)

Medina (2024) describe el Software Architecture Analysis Method (SAAM) es una técnica estructurada para evaluar arquitecturas de software mediante el análisis de escenarios que representan posibles usos, cambios o interacciones futuras del sistema. Su propósito es determinar en qué medida la arquitectura satisface atributos de calidad como la modificabilidad, interoperabilidad, disponibilidad y cobertura funcional. Este enfoque permite identificar fortalezas, debilidades y riesgos antes de la implementación, facilitando decisiones informadas sobre el diseño arquitectónico. Esta metodología ha sido destacada por su aplicabilidad práctica tanto en contextos académicos como industriales.

2.1.15. Métricas de evaluación automática en generación de texto

La evaluación de modelos de lenguaje de gran escala (LLMs) requiere métricas que permitan analizar tanto la similitud estructural como la cercanía semántica entre el texto generado y una referencia humana. En este trabajo se emplearon ROUGE-L y similitud coseno sobre embeddings, en línea con prácticas ampliamente documentadas en la literatura reciente sobre evaluación de generación textual (Gao et al., 2022; Zhang et al., 2023).

ROUGE-L ROUGE (Recall-Oriented Understudy for Gisting Evaluation) es un conjunto de métricas diseñadas para comparar automáticamente textos generados frente a referencias humanas. En particular, ROUGE-L se basa en el cálculo de la *Longest Common Subsequence* (LCS), lo que permite capturar similitud estructural incluso cuando no existen coincidencias exactas de n-gramas contiguos (Lin, 2004).

Aunque fue propuesta originalmente en el contexto de resumen automático, investigaciones recientes continúan utilizándola como métrica base para evaluación estructural en modelos de lenguaje contemporáneos (Gao et al., 2022; Zhang et al., 2023). Su permanencia en la literatura se debe a su capacidad de ofrecer una medida reproducible de alineación textual frente a una referencia.

Similitud coseno sobre embeddings La similitud coseno es una medida geométrica que cuantifica el ángulo entre dos vectores en un espacio vectorial. En el contexto del procesamiento moderno de lenguaje natural, estos vectores corresponden a embeddings densos generados por modelos neuronales.

El uso de embeddings para evaluación semántica se ha consolidado como estándar en benchmarks recientes de modelos de lenguaje, dado que permite capturar cercanía semántica incluso cuando la redacción superficial difiere (Reimers and Gurevych, 2019; Muennighoff et al., 2023). Esta aproximación resulta particularmente relevante en la evaluación de LLMs, donde variaciones estilísticas no necesariamente implican divergencias conceptuales.

La combinación de ROUGE-L y similitud coseno permite realizar una evaluación complementaria: la primera orientada a coincidencia estructural y la segunda a proximidad semántica, siguiendo recomendaciones actuales sobre evaluación multidimensional de modelos generativos (Gao et al., 2022).

2.1.16. Architecture Tradeoff Analysis Method (ATAM) y su variante ATAM-lite

El método ATAM (*Architecture Tradeoff Analysis Method*) fue desarrollado por el Software Engineering Institute (SEI) de Carnegie Mellon University como una técnica estructurada y basada en escenarios para evaluar arquitecturas de software frente a múltiples atributos de calidad en competencia —rendimiento, seguridad, modificabilidad, disponibilidad, entre otros— identificando puntos de sensibilidad, puntos de compromiso (*tradeoff points*) y riesgos arquitectónicos (Kazman et al., 1998, 2000). En su forma completa, el ATAM requiere sesiones de evaluación de tres a cuatro días con equipos externos, representantes de stakeholders y arquitectos (Kazman et al., 2000).

La variante denominada *ATAM-lite* es una adaptación ligera para contextos donde los recursos, los plazos o el tamaño del equipo no permiten ejecutar el ATAM completo. Conserva la esencia del método —identificación de escenarios, análisis de decisiones arquitectónicas y detección de riesgos— reduciendo el número de pasos formales y prescindiendo de artefactos más elaborados como el árbol de utilidad completo (Nganga, 2025).

2.2. Estado del Arte

2.2.1. Antecedentes (trabajos previos)

A continuación, se presentan los antecedentes que se consideran más significativos:

- Plataforma tecnológica para habilitar la venta consultiva de un producto digital con inteligencia artificial (Zambrano Barco, 2024). Este trabajo fue desarrollado en el marco del programa de Maestría en Ingeniería de Software de la Pontificia Universidad Javeriana Cali, El proyecto propone una solución arquitectónica para la adopción de un modelo de inteligencia artificial que actúe como un trabajador digital inteligente, capaz de mantener conversaciones consultivas orientadas a la venta de productos digitales. Se enfoca en el diseño y despliegue de una arquitectura de software basada en servicios en la nube (AWS), habilitada para integrar múltiples modelos de procesamiento de lenguaje natural (NLP). El documento aborda el problema de la falta de integración y rendimiento en sistemas que adoptan asistentes digitales, y plantea una solución robusta mediante técnicas modernas de diseño arquitectónico y automatización conversacional.

  - Aplicación de inteligencia artificial en procesos de venta consultiva: Se desarrolla un trabajador digital que sostiene conversaciones a través de WhatsApp, utilizando modelos NLP para ofrecer asesoría automatizada y personalizada sobre productos financieros.
  - Diseño de arquitectura basada en atributos de calidad (ADD): Utiliza la metodología ADD para estructurar iterativamente la arquitectura, priorizando atributos como seguridad, fiabilidad e interoperabilidad. Este enfoque permite garantizar la calidad no funcional del sistema.
  - Orquestación del flujo conversacional con RAG: La técnica Retrieval-Augmented Generation (RAG) es empleada para enriquecer las respuestas del trabajador digital, utilizando datos actualizados extraídos de una base de conocimiento gestionada por una base de datos vectorial (Pinecone).
  - Infraestructura tecnológica desplegada en la nube: El sistema fue implementado sobre servicios como Amazon ECS, Bedrock, DynamoDB y S3, y complementado con Flowise AI como plataforma Low-Code/No-Code para la orquestación de flujos conversacionales.
  - Evaluación mediante metodologías arquitectónicas: La arquitectura fue evaluada utilizando la metodología LAE (Lightweight Architecture Evaluation), analizando atributos de calidad clave mediante escenarios y técnicas estructuradas.

  Este trabajo representa un caso relevante de aplicación de inteligencia artificial y arquitectura de software para la automatización de procesos de atención y venta, sirviendo como referencia para soluciones futuras en sectores donde la interacción consultiva automatizada sea estratégica

- Framework para la integración de herramientas de Inteligencia Artificial en los productos de software para el área de Seguridad y Salud en el Trabajo desarrollados por la Corporación Talentum (Cuellar, 2024) Fue desarrollado en la Pontificia Universidad Javeriana Cali, como parte de la Maestría en Ingeniería de Software. Su objetivo fue diseñar un framework que permitiera integrar componentes de Inteligencia Artificial (IA) en sistemas de software existentes, específicamente en el área de Seguridad y Salud en el Trabajo (SST), utilizando como caso de estudio la Corporación Talentum.

- Desarrollo de un framework basado en buenas prácticas: Se identificaron prácticas óptimas para la integración de IA, considerando aspectos como selección de modelos LLM, estructuración de datos, uso de bases vectoriales y evaluación de prompts.
- Diseño de arquitectura con Attribute-Driven Design (ADD): El marco fue diseñado utilizando la metodología ADD, priorizando atributos como interoperabilidad y rendimiento, y fue complementado con representaciones en modelos UML y C4.
- Aplicación de RAG y bases vectoriales: Se implementó arquitectura RAG sobre servicios como Flowise, DynamoDB y Qdrant, logrando orquestar la carga, vectorización, almacenamiento y consulta de información textual asociada a matrices de riesgo en SST.
- Ingeniería de prompts adaptada al dominio SST: Se elaboraron criterios específicos para mejorar la precisión de las respuestas generadas por la IA, usando prompts contextualmente definidos y validados por especialistas del dominio.
- Validación con especialistas y pruebas de arquitectura ligera (LAE): Se construyó un prototipo funcional que fue validado mediante escenarios reales, análisis de calidad arquitectónica, y retroalimentación de usuarios expertos en SST.

Este trabajo resulta altamente relevante para soluciones que integran IA en contextos especializados, como el comercial o consultivo, y aporta una base metodológica aplicable a múltiples sectores donde la precisión contextual y la integración con sistemas existentes son claves.

- An Open-Source RAG Architecture for LLMs (Boppana et al., 2024). El documento describe y brinda información sobre el uso de Large Language Models (LLMs) en combinación con bases de datos vectoriales dentro de una arquitectura Retrieval-Augmented Generation (RAG). Este modelo aborda problemas como las alucinaciones del modelo, la dependencia de datos desactualizados y la falta de conocimiento especializado en ciertos dominios. En el artículo se describen varios puntos que se consideran necesarios resaltar:

  - Importancia del enfoque RAG: El modelo RAG propuesto combina la generación de texto con la recuperación de información en tiempo real, mejorando la precisión y reduciendo la generación de respuestas incorrectas en los LLMs.
  - Uso de bases de datos vectoriales: Se resalta la integración de ChromaDB en una instancia de AWS EC2 para almacenar y recuperar información semánticamente relevante, optimizando la precisión en la generación de contenido.
  - Infraestructura en la nube: La arquitectura está desplegada en Amazon Web Services (AWS), utilizando servicios como AWS Textract para la extracción de información de documentos y AWS Lambda para la automatización de flujos de trabajo. Este enfoque permite escalabilidad y optimización del rendimiento computacional.
  - Impacto en la automatización: La combinación de recuperación y generación de información mejora la calidad del contenido generado, permitiendo a los modelos LLM proporcionar respuestas más precisas en entornos donde la actualización constante de datos es esencial.

  Este documento representa un aporte clave en el desarrollo de modelos híbridos que integran generación y recuperación de información, siendo relevante para aplicaciones en automatización de procesos comerciales, motores de búsqueda y categorización de productos

- Plan de negocio para la creación de Turismo IA (Montoya and Burbano, 2023). Empresa con enfoque en la prestación de servicios de inteligencia artificial para el sector del turismo. Este trabajo propone la creación de una empresa especializada en inteligencia artificial aplicada al turismo, con el objetivo de mejorar la atención al cliente, optimizar la personalización de experiencias y aumentar la eficiencia operativa en agencias de viajes y otros actores del sector. El documento explora la integración de chatbots inteligentes, automatización de procesos y generación de contenido con IA para transformar la industria turística.

  - Aplicación de inteligencia artificial en la atención al cliente: La empresa Turismo IA propone el uso de chatbots avanzados con IA generativa para gestionar interacciones con clientes en agencias de viaje, proporcionando respuestas rápidas y automatizadas sin intervención humana.
  - Personalización de experiencias turísticas: Se busca utilizar modelos de procesamiento de lenguaje natural para adaptar las recomendaciones y servicios turísticos según las preferencias de cada cliente, mejorando la satisfacción del usuario.
  - Optimización de la competitividad en el sector: La implementación de IA no solo automatiza procesos, sino que también ayuda a las empresas turísticas a ser más competitivas en el mercado digital mediante análisis de datos y generación de contenido publicitario automatizado
  - Infraestructura tecnológica basada en la nube: El estudio presenta un enfoque en soluciones SaaS (Software as a Service) para ofrecer servicios de chatbot y análisis de datos a empresas del sector turismo, eliminando la necesidad de infraestructura física costosa y garantizando escalabilidad.
  - Impacto en la transformación digital del turismo: Se enfatiza cómo la IA puede revolucionar la industria turística en Colombia, brindando herramientas accesibles para pequeñas y medianas agencias de viajes que de otro modo no tendrían acceso a estas tecnologías.

  Este trabajo representa un avance significativo en la exploración de la inteligencia artificial aplicada a la industria del turismo, sirviendo como un referente clave para la integración de tecnologías emergentes en modelos de negocio innovadores.

- Generative AI for Sales: A Study on the Potential of Retrieval Augmented Generation for Response Automation in the Request for Proposal Process in Telecommunications (Bilal and Åström, 2024).Este estudio analiza el uso de Generative AI (GenAI) y Retrieval Augmented Generation (RAG) en la automatización del proceso de respuesta a solicitudes de propuestas (RFPs) dentro del sector de telecomunicaciones. La investigación presenta un Proof of Concept (POC) para evaluar cómo la inteligencia artificial generativa puede mejorar la eficiencia en la generación de respuestas, reduciendo el tiempo y los costos asociados a este proceso crítico en ventas B2B.

  En el documento se resaltan los siguientes puntos clave:

  - Integración de RAG en la automatización de respuestas: Se explora cómo la combinación de modelos generativos y recuperación de información específica mejora la precisión de las respuestas, mitigando problemas como las alucinaciones de los modelos LLM y la falta de conocimiento especializado en estos sistemas.

- Evaluación de estrategias de segmentación de texto (chunking): El estudio prueba diferentes formas de dividir la información para mejorar la recuperación de datos relevantes en RAG. Se concluye que fragmentos más pequeños funcionan mejor para preguntas simples, mientras que segmentos más grandes son más adecuados para consultas técnicas complejas
- Análisis de rendimiento de modelos de IA en RFPs: La evaluación de LLaMA 2 y otras soluciones abiertas indica que los modelos pueden automatizar la creación de respuestas con alta precisión. Sin embargo, se identifican desafíos en el manejo de datos confidenciales y la necesidad de estructuración eficiente de bases de conocimiento.
- Aplicaciones en la optimización del ciclo de ventas: Se destaca el impacto de RAG en la reducción de tiempos de respuesta en telecomunicaciones, permitiendo a las empresas mejorar su competitividad en licitaciones y proyectos de alto valor.
- Desafíos y limitaciones: Aunque la IA generativa ofrece ventajas en la automatización de respuestas, su integración en entornos empresariales debe considerar aspectos de seguridad de datos, adaptación a dominios específicos y compatibilidad con sistemas existentes.

Este documento representa un avance significativo en la investigación sobre la aplicación de IA generativa en la automatización de procesos comerciales, proporcionando una base sólida para futuras optimizaciones en el uso de RAG en ventas y telecomunicaciones.

2.2.2. Análisis comparativo de los antecedentes

Los trabajos revisados coinciden en el uso de arquitecturas RAG y modelos de lenguaje para automatizar procesos que requieren intervención humana especializada. La viabilidad de estas arquitecturas en contextos consultivos y de gestión documental ha sido demostrada mediante metodología ADD y evaluación LAE (Zambrano Barco, 2024; Cuellar, 2024), mientras que otros trabajos aportan evidencia técnica sobre su aplicación en procesos comerciales y la reducción de alucinaciones mediante RAG (Bilal and Åström, 2024; Boppana et al., 2024). La automatización de procesos de atención en sectores de servicios mediante IA generativa ha sido explorada también en contextos como el turismo (Montoya and Burbano, 2023).

Sin embargo, estos antecedentes abordan la IA generativa principalmente desde la implementación de prototipos funcionales, sin profundizar en el diseño arquitectónico formal ni en la evaluación sistemática de atributos de calidad. Ninguno contempla la integración multivendor de modelos de IA ni la portabilidad del diseño entre entornos de despliegue. El presente trabajo cubre estas brechas mediante una arquitectura evaluada con SAAM y ATAM-lite (Kazman et al., 2000), una estrategia de enrutamiento dinámico multivendor y validación en entornos complementarios Docker y AWS.

Capítulo 3

Análisis y diagnóstico

3.1. Fases metodológicas del proyecto

El desarrollo del proyecto se estructuró en un conjunto de fases metodológicas que permiten cumplir de manera ordenada los objetivos propuestos. Cada fase agrupa actividades específicas orientadas al análisis, diseño, documentación, validación y presentación de los resultados obtenidos. Estas actividades se alinean directamente con los objetivos general y específicos definidos en la sección correspondiente, garantizando trazabilidad entre las tareas ejecutadas y los resultados esperados. En la siguiente tabla se presenta la relación detallada entre las fases, las actividades realizadas y los objetivos asociados que sustentan el cumplimiento del propósito del proyecto.

Fase	**Actividad**	**Objetivo asociado (referencia interna)**
Investigación y diagnóstico	Investigación de modelos arquitectónicos existentes aplicables a la automatización de ofertas comerciales.	OE1 (*ver Objetivo Específico **OE1:***)
Investigación y diagnóstico	Análisis de requerimientos de Infraestructura Virtual SAS para definir necesidades específicas.	OE1 (*ver Objetivo Específico **OE1:***)
Investigación y diagnóstico	Análisis de soluciones multivendor para IA generativa tipo RAG.	OE3 (*ver Objetivo Específico **OE3:***)
Diseño arquitectónico	Elaboración del Diagrama de Contexto (Modelo C4).	OE2 (*ver Objetivo Específico **OE2:***)
Diseño arquitectónico	Revisión y validación con el director.	OG (*ver Objetivo General 1.3.1*)
Diseño arquitectónico	Identificación de componentes tecnológicos en AWS que se integrarán en la solución.	OE2 (*ver Objetivo Específico **OE2:***)
Diseño arquitectónico	Diseño del Diagrama de Contenedores (Modelo C4) para representar los principales módulos de la arquitectura.	OE2 (*ver Objetivo Específico **OE2:***)
Diseño arquitectónico	Definición de patrones arquitectónicos adecuados para la solución.	OE2 (*ver Objetivo Específico **OE2:***)

Fase	Actividad	Objetivo asociado (referencia interna)
Diseño arquitectónico	Diseño del componente Hub de IA para integración desacoplada vía APIs.	OE3 (*ver Objetivo Específico **OE3:***)
Diseño arquitectónico	Definición de estrategia de enrutamiento dinámico de modelos (AI model routing).	OE3 (*ver Objetivo Específico **OE3:***)
Diseño arquitectónico	Elaboración del Diagrama de Componentes (Modelo C4).	OE2 (*ver Objetivo Específico **OE2:***)
Documentación y evaluación de calidad	Documentación de los atributos de calidad que se evaluarán en la solución.	OE5 (*ver Objetivo Específico **OE5:***)
Documentación y evaluación de calidad	Revisión y validación con el director.	OE5 (*ver Objetivo Específico **OE5:***)
Documentación y evaluación de calidad	Revisión de cada componente de la arquitectura en relación con los atributos de calidad definidos.	OE2 (*ver Objetivo Específico **OE2:***)
Documentación y evaluación de calidad	Evaluación de requisitos de seguridad, rendimiento y fiabilidad.	OE2 (*ver Objetivo Específico **OE2:***)
Documentación y evaluación de calidad	Análisis de riesgos y mitigación de problemas potenciales en la arquitectura.	OE5 (*ver Objetivo Específico **OE5:***)
Documentación y evaluación de calidad	Ajuste de la documentación técnica con base en la validación realizada.	OE5 (*ver Objetivo Específico **OE5:***)
Documentación y evaluación de calidad	Generación del informe de evaluación de calidad de la arquitectura.	OE5 (*ver Objetivo Específico **OE5:***)
PoC (Prueba de concepto)	Preparación del ambiente para la PoC.	OE4 (*ver Objetivo Específico **OE4:***)
PoC (Prueba de concepto)	Implementación de prueba de concepto (PoC) para generación automática de propuestas.	OE4 (*ver Objetivo Específico **OE4:***)
PoC (Prueba de concepto)	Validación funcional y comparativa del PoC vs. proceso manual.	OE4 (*ver Objetivo Específico **OE4:***)
PoC (Prueba de concepto)	Documentación del PoC y análisis de eficiencia.	OE4 (*ver Objetivo Específico **OE4:***)

Fase	Actividad	Objetivo asociado (referencia interna)
Presentación y cierre	Refinamiento de los diagramas del modelo C4 según observaciones previas.	OE5 (*ver Objetivo Específico **OE5:***)
Presentación y cierre	Revisión final del documento del proyecto para verificar cumplimiento de los objetivos.	OG (*ver Objetivo General 1.3.1*)
Presentación y cierre	Preparación de la presentación del trabajo.	OG (*ver Objetivo General 1.3.1*)
Presentación y cierre	Comparación de la solución propuesta con el proceso manual actual.	OE4 (*ver Objetivo Específico **OE4:***)
Presentación y cierre	Entrega del informe final y presentación de resultados.	OE5 (*ver Objetivo Específico **OE5:***)

Nota: OG y OE1–OE5 corresponden a los objetivos documentados en la Sección 1.3 del presente trabajo.

3.2. Modelos arquitectónicos para la automatización de ofertas comerciales

Un modelo arquitectónico es una representación abstracta que define la estructura, los componentes y las interacciones de un sistema de software, orientando las decisiones de diseño para cumplir con requisitos funcionales y atributos de calidad específicos (Bass et al., 2012). En el marco de esta investigación, el análisis de modelos arquitectónicos existentes permite identificar las estructuras y patrones más adecuados para la automatización del proceso de generación de ofertas comerciales, garantizando seguridad, rendimiento, fiabilidad y escalabilidad conforme a la norma ISO/IEC 25010. Esta revisión sustenta además la incorporación de tecnologías emergentes —como la inteligencia artificial generativa y los mecanismos de recuperación aumentada (RAG)— en una arquitectura flexible basada en la nube.

A continuación, se presentan los modelos arquitectónicos existentes y la justificación de su aplicación:

3.2.1. Arquitectura en Capas (Layered Architecture)

La arquitectura en capas constituye uno de los estilos más tradicionales en la ingeniería de software. Propone la separación del sistema en niveles jerárquicos con responsabilidades bien definidas, generalmente compuestos por capa de presentación, capa de negocio, capa de servicios y capa de datos (Bass et al., 2012; Fowler, 2003).

Aplicación en la tesis: En el contexto de la automatización de ofertas comerciales, este modelo permite establecer una estructura ordenada donde las interfaces de usuario (formularios de preventa) se desacoplan de la lógica de generación de propuestas, de los servicios de inteligencia artificial y de la persistencia de datos. Esta división facilita el mantenimiento, la trazabilidad de cambios y la seguridad de la información, al restringir el acceso entre capas.

Contribución: Favorece la mantenibilidad, seguridad y fiabilidad del sistema, cumpliendo el objetivo de definir elementos que soporten los atributos de calidad establecidos en ISO 25010.

3.2.1.1. Arquitectura Orientada a Servicios (Service-Oriented Architecture, SOA)

SOA se basa en la creación de servicios independientes que se comunican mediante interfaces estandarizadas, lo que facilita la interoperabilidad entre sistemas heterogéneos (Erl, 2005).

Aplicación en la tesis: El proceso de preventa puede representarse como un conjunto de servicios: análisis de requerimientos, cálculo de precios, validación técnica, aprobación comercial y generación de documentos. Cada servicio puede implementarse de manera independiente y exponerse mediante APIs, lo que permite que otras aplicaciones, como el CRM o el sistema de IA, interactúen sin acoplamiento directo.

Contribución: Promueve la interoperabilidad, la modularidad y la reutilización, atributos esenciales para una arquitectura flexible que permita integrar múltiples mecanismos de IA mediante APIs, en coherencia con el objetivo OE3.

3.2.1.2. Arquitectura de Microservicios (Microservices Architecture)

La arquitectura de microservicios es una evolución de SOA que descompone el sistema en servicios pequeños, autónomos y desplegables de forma independiente. Cada microservicio encapsula una funcionalidad específica y se comunica con los demás mediante un API Gateway (Newman, 2015; Francesco et al., 2019).

Aplicación en la tesis: El flujo de automatización de ofertas puede dividirse en microservicios especializados: gestión de oportunidades, análisis técnico, motor de reglas de precios, generación de texto con IA, validación humana y almacenamiento de resultados. Este enfoque permite escalar de manera independiente los componentes que demandan mayor carga computacional, como el módulo de IA o el generador de documentos.

Contribución: Aumenta la escalabilidad, el rendimiento y la fiabilidad del sistema, al aislar fallos y facilitar el despliegue progresivo en entornos de nube. Cumple con el objetivo OE2 al maximizar atributos de calidad y con el OE3 al permitir una integración dinámica con servicios de IA.

3.2.1.3. Arquitectura Basada en Eventos (Event-Driven Architecture, EDA)

EDA propone un modelo asincrónico donde los componentes del sistema se comunican a través de eventos que desencadenan acciones específicas. Este estilo es ampliamente utilizado en sistemas distribuidos que requieren reactividad y desacoplamiento (Chandy et al., 2010).

Aplicación en la tesis: El proceso comercial puede modelarse como un conjunto de eventos, tales como “nueva solicitud de cotización”, “validación aprobada” o “documento generado”. Cada evento activa automáticamente los servicios correspondientes sin requerir una secuencia rígida.

Contribución: Favorece la escalabilidad y la fiabilidad, ya que el sistema continúa operando incluso si un componente falla temporalmente. Además, mejora el rendimiento al permitir procesamiento paralelo y asincrónico. Su adopción contribuye directamente al objetivo OE2 y se integra con la infraestructura cloud-native prevista en el OE3.

3.2.1.4. Arquitectura Nativa en la Nube (Cloud-Native Architecture)

La arquitectura nativa en la nube se basa en el uso de servicios gestionados y componentes elásticos que se adaptan automáticamente a la demanda. Su propósito es garantizar resiliencia, portabilidad y escalabilidad (Fehling et al., 2014).

Aplicación en la tesis: La solución puede implementarse sobre plataformas como AWS o Azure utilizando servicios administrados (por ejemplo, Lambda, API Gateway, DynamoDB o Azure Functions), permitiendo elasticidad sin intervención manual. Este enfoque reduce los tiempos de despliegue, mejora la disponibilidad y optimiza los costos operativos.

Contribución: Maximiza la escalabilidad y la eficiencia de recursos, cumpliendo los objetivos de diseñar una arquitectura flexible basada en nube y optimizar el rendimiento bajo demanda (OE3).

3.2.1.5. Arquitectura Serverless

El modelo serverless elimina la gestión directa de servidores, ejecutando funciones bajo demanda en entornos totalmente gestionados (Baldini et al., 2017).

Aplicación en la tesis: Tareas puntuales del proceso de preventa, como la generación de un documento de oferta, la inferencia de IA o la validación de datos, pueden ejecutarse mediante funciones serverless. Esto permite una operación altamente escalable y económica, dado que el cómputo se activa únicamente cuando es necesario.

Contribución: Optimiza el rendimiento y la escalabilidad, reduce la carga administrativa y mejora la fiabilidad mediante la gestión automática de recursos por parte del proveedor cloud. Su adopción es coherente con los principios de arquitectura cloud-native y con los objetivos OE2 y OE3.

3.2.1.6. Arquitectura Centrada en Datos (Data-Centric / Knowledge-Centric Architecture)

Este modelo organiza el sistema alrededor de la gestión y explotación del conocimiento, considerando los datos como el principal activo estratégico (Inmon, 2005).

Aplicación en la tesis: En el contexto de Infraestructura Virtual SAS, las propuestas comerciales, fichas técnicas y documentos históricos representan una fuente valiosa de conocimiento que puede ser sistematizada y reutilizada. Al implementar un repositorio centralizado de información estructurada y no estructurada, la arquitectura facilita la consulta y el entrenamiento de modelos de IA.

Contribución: Aporta fiabilidad y mantenibilidad al garantizar la consistencia y trazabilidad de la información reutilizada. Además, soporta directamente los mecanismos de recuperación utilizados por el modelo RAG, fortaleciendo el cumplimiento del objetivo OE3.

3.2.1.7. Arquitectura RAG (Retrieval-Augmented Generation)

La arquitectura RAG combina técnicas de recuperación de información (retrieval) con modelos generativos (LLMs) para producir respuestas fundamentadas en fuentes documentales verificables (Lewis et al., 2020; Zhao et al., 2024).

Aplicación en la tesis: Este modelo constituye el núcleo de la capa de inteligencia artificial del sistema. Permite que la generación de propuestas se base en información histórica, plantillas corporativas

y políticas comerciales previamente validadas. De esta manera, la IA genera contenido coherente, específico y alineado con los estándares de la empresa.

Contribución: Fortalece la fiabilidad y la seguridad del contenido generado, al reducir errores y garantizar trazabilidad de las fuentes utilizadas. Su implementación mediante APIs cumple plenamente el objetivo OE3, al habilitar la integración de múltiples mecanismos de IA en una arquitectura flexible.

3.2.1.8. Arquitectura de Flujo de Procesamiento o Pipeline Architecture

El modelo de arquitectura tipo pipeline organiza el sistema en una secuencia de etapas independientes por las cuales los datos fluyen de manera estructurada: ingesta, transformación, análisis y salida (Gorton, 2011).

Aplicación en la tesis: El flujo de generación de ofertas puede modelarse como una cadena de procesamiento: ingreso de requerimientos, validación, búsqueda documental, generación, revisión y publicación. Cada etapa opera como un módulo autónomo que transforma la información y la transmite a la siguiente.

Contribución: Facilita la trazabilidad, mantenibilidad y rendimiento, al permitir el control de cada fase del proceso comercial. Es complementaria a la arquitectura basada en eventos y apoya directamente el objetivo OE2.

3.2.1.9. Arquitectura de Orquestación de Modelos o AI Gateway (Multi-LLM Architecture)

La arquitectura AI Gateway, también conocida como multi-LLM orchestration, define una capa intermedia que unifica el acceso y la gestión de múltiples modelos de lenguaje o servicios de inteligencia artificial, aplicando políticas comunes de autenticación, auditoría, monitoreo y enrutamiento (OpenAI, 2023b; Hugging Face, 2024; Amazon Web Services, 2025).

Aplicación en la tesis: Este modelo permite que la solución utilice diferentes modelos de IA (GPT, Claude, Gemini, Llama, Titan, entre otros) seleccionando dinámicamente el más adecuado según la tarea, el costo o el nivel de precisión requerido. Asimismo, proporciona un control centralizado sobre la seguridad de los datos y el cumplimiento normativo.

Contribución: Garantiza portabilidad, seguridad y fiabilidad, al evitar dependencia de un único proveedor y permitir continuidad operativa ante fallas de un modelo. Constituye la base para lograr la integración multivendor prevista en el objetivo OE3.

3.3. Matriz de Decisión MLLs.

Ponderado:

La asignación de los criterios y sus respectivos pesos dentro de la matriz de decisión se fundamentó en los resultados obtenidos a partir de una encuesta aplicada a especialistas y actores del proceso de preventa de Infraestructura Virtual SAS, complementada con los atributos de calidad establecidos por la norma ISO/IEC 25010. La encuesta, implementada en Microsoft Forms, permitió recopilar percepciones sobre la relevancia de cada criterio —rendimiento, costo, seguridad y compliance, escalabilidad, eficiencia energética y claridad e interpretabilidad— utilizando una escala de Likert de 1 (poco importante) a 5 (muy importante).

Los resultados evidenciaron una clara priorización del rendimiento y la eficiencia (4.7/5) como el factor más determinante, seguido de seguridad y compliance (4.5/5), costo (4.3/5) y escalabilidad (4.1/5). Los criterios de claridad e interpretabilidad (3.9/5) y eficiencia energética (3.7/5), aunque con menor ponderación relativa, se mantuvieron dentro de los aspectos relevantes por su impacto en la transparencia y sostenibilidad tecnológica.

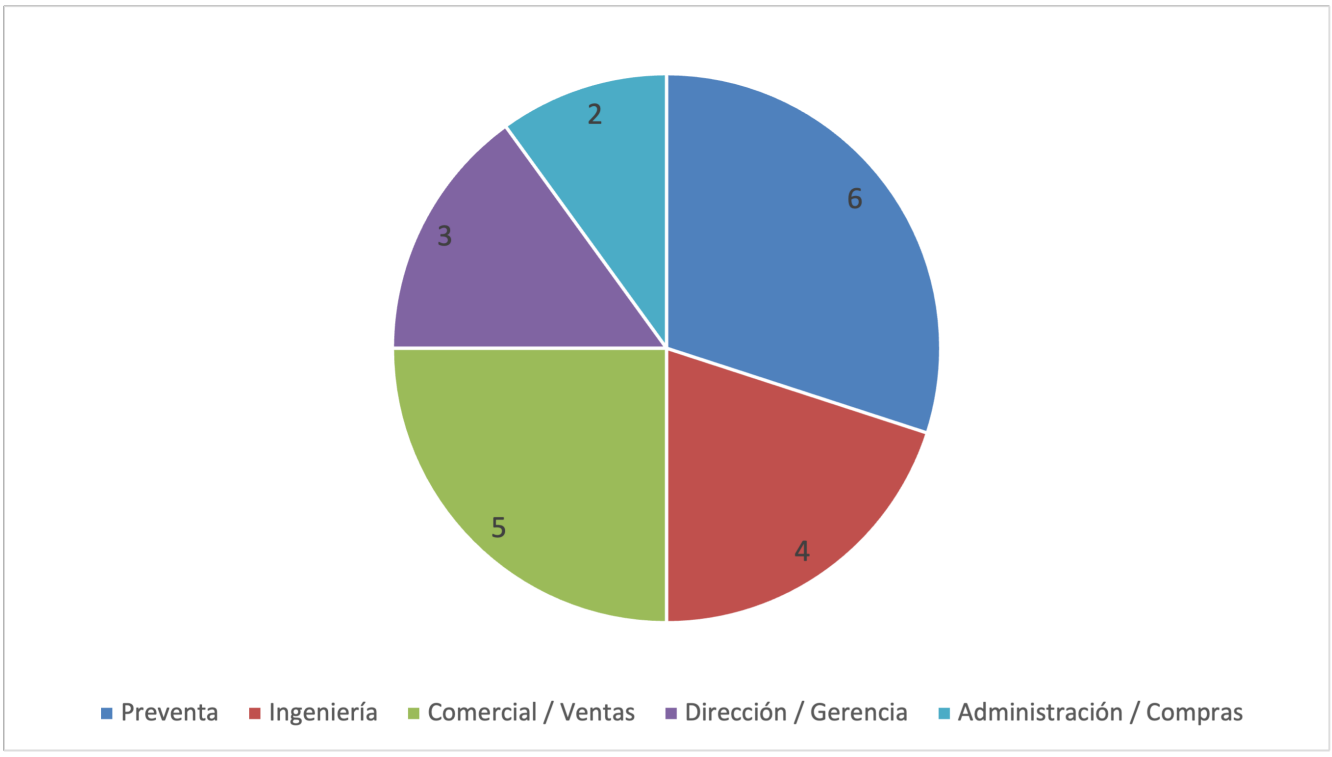


Figura 3.1: Resultado por área

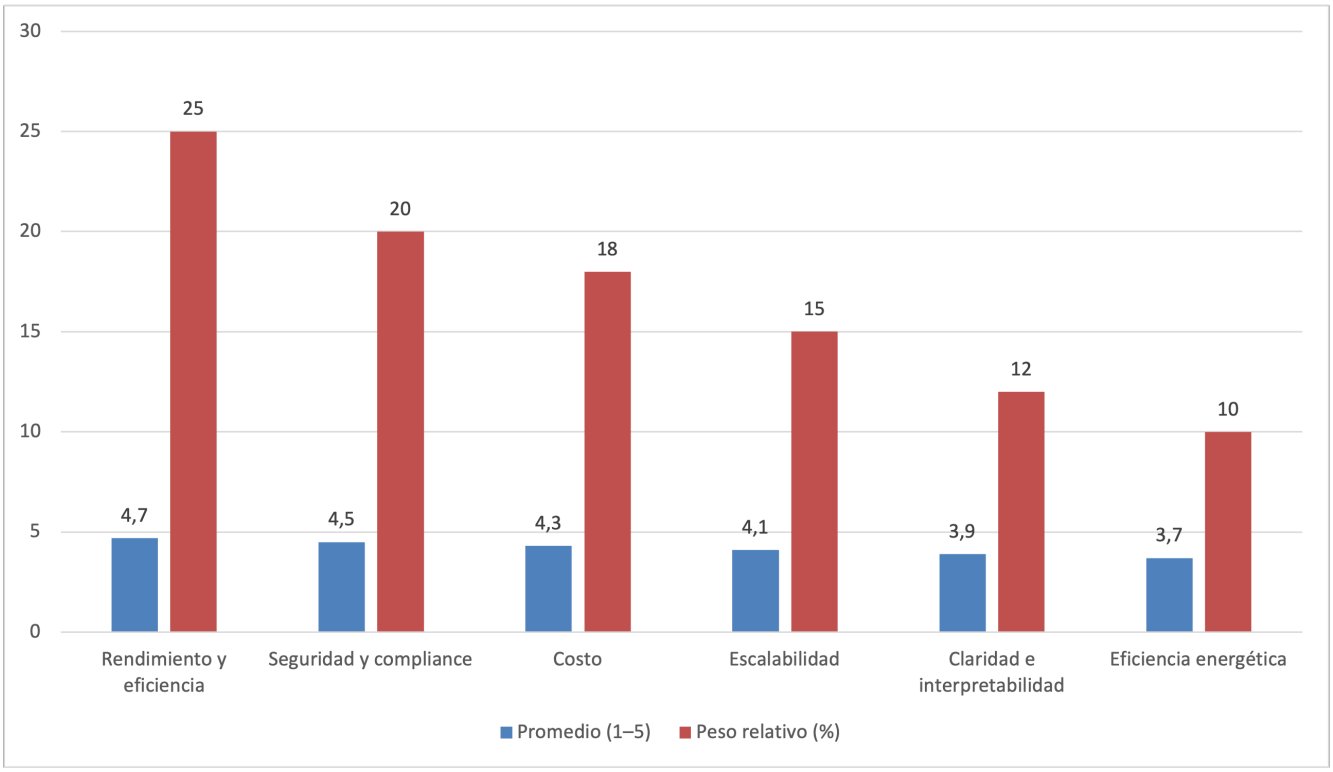


Figura 3.2: Resultado encuesta

A partir de estos resultados se definieron los pesos normalizados empleados en la matriz (Figure 3.3), asegurando una ponderación basada en evidencia empírica y alineada con los atributos de calidad más valorados por los participantes. Este enfoque combinó fundamentos normativos (ISO/IEC 25010) con una validación participativa, fortaleciendo la justificación técnica y metodológica del modelo de selección de LLMs.

Los pesos se convirtieron a su equivalente decimal de la siguiente manera:

- Rendimiento: 25 % → 0.25
- Seguridad / Compliance: 20 % → 0.20
- Costo: 18 % → 0.18
- Escalabilidad: 15 % → 0.15
- Interpretabilidad: 12 % → 0.12
- Eficiencia energética: 10 % → 0.10

La fórmula general empleada fue:

$$\text{Ponderado} = (R \times 0{,}25) + (S \times 0{,}20) + (C \times 0{,}18) + (E \times 0{,}15) + (I \times 0{,}12) + (En \times 0{,}10)$$

donde:

- R = valor del criterio **Rendimiento** (escala 1–5)
- C = valor del criterio **Costo** (escala 1–5)
- S = valor del criterio **Seguridad / Compliance** (escala 1–5)
- E = valor del criterio **Escalabilidad** (escala 1–5)
- En = valor del criterio **Eficiencia energética** (escala 1–5)
- I = valor del criterio **Interpretabilidad** (escala 1–5)

Justificación de la asignación de pesos a los criterios de evaluación:

- **Rendimiento (25 %)**. Se definió como el criterio de mayor peso porque impacta directamente en la eficiencia del proceso de generación de propuestas. Un modelo con buen desempeño permite reducir significativamente los tiempos de respuesta y procesar información técnica y comercial de manera ágil, lo cual es clave en el entorno de preventa.
- **Costo (18 %)**. Ocupa el segundo lugar, ya que la sostenibilidad económica del sistema depende en gran medida del valor por uso de cada modelo. Se priorizan alternativas que mantengan una relación costo–beneficio adecuada sin comprometer la calidad de los resultados.

  Nota: Se establece en escala inversa, donde valores mayores representan menor costo (más económico) y valores menores, mayor costo (más caro por millón de tokens o licenciamiento).
- **Seguridad y Compliance (20 %)**. Este criterio garantiza que los modelos seleccionados cumplan con estándares de confidencialidad y manejo responsable de datos, considerando que las propuestas comerciales incluyen información sensible de clientes y precios.
- **Escalabilidad (15 %)**. Se incluye para evaluar la capacidad del modelo de integrarse en una arquitectura flexible, multivendor y desplegable en diferentes entornos cloud, asegurando que la solución pueda evolucionar y adaptarse a mayores demandas futuras.
- **Eficiencia energética (10 %)**. Responde al compromiso con la sostenibilidad tecnológica y el uso responsable de recursos computacionales, en coherencia con los Objetivos de Desarrollo Sostenible (ODS 8 y 9).
- **Interpretabilidad (12 %)**. Se considera esencial para garantizar la transparencia de los resultados generados por la inteligencia artificial. Este aspecto permite que los equipos humanos comprendan y validen las respuestas, reforzando la confianza en el sistema y su aplicabilidad real en procesos de negocio.

Modelo	Fabricante	RAG	Rendimiento	Costo	Seguridad	Escalabilidad	Energía	Interpretabilidad	Ponderado
GPT-5	OpenAI	Sí	5	2	5	5	4	5	4,36
GPT-5 mini	OpenAI	Sí	4	5	5	5	4	4	4,53
GPT-4o	OpenAI	Sí	5	3	5	5	4	4	4,42
Claude 4.5 Sonnet	Anthropic	Sí	5	3	5	5	4	5	4,54
Claude 3.5 Sonnet	Anthropic	Sí	5	3	5	5	4	5	4,54
Gemini 1.5 Pro	Google DeepMind	Sí	5	3	5	5	5	5	4,64
Gemini 2.0 Flash	Google DeepMind	Sí	4	4	5	5	5	4	4,45
Titan Text / Nova Text	Amazon AWS	Sí	4	4	5	5	4	4	4,35
Titan Embeddings v2	Amazon AWS	Sí	4	4	5	5	5	4	4,45
Llama 3.1 (8B/70B/405B)	Meta	Sí	4	5	3	4	4	4	3,98
Llama 4 (Scout/Maverick)	Meta	Sí	5	4	3	4	4	4	4,05
Mistral Large 2	Mistral	Sí	4	4	4	5	5	4	4,25
Mixtral 8x7B / Mistral 7B	Mistral	Sí	4	5	3	4	5	3	3,96
Cohere Command R+	Cohere	Sí	4	5	4	4	4	5	4,3
Cohere Command R	Cohere	Sí	4	5	4	4	4	4	4,18
DeepSeek V3 / Coder V2	DeepSeek AI	Sí	4	5	3	4	4	3	3,86
Grok-3 / Grok-2	xAI	Sí	4	4	4	4	4	3	3,88

Figura 3.3: Tabla Matriz de Decisión MLLs

3.4. Análisis de soluciones multivendor para IA generativa tipo RAG.

1. Ecosistema de herramientas, plataformas y frameworks multivendor.

   La adopción de inteligencia artificial generativa en entornos empresariales ha impulsado el surgimiento de soluciones multivendor que permiten integrar múltiples modelos de lenguaje (LLMs) bajo una arquitectura desacoplada. Este enfoque resulta especialmente relevante en arquitecturas tipo Retrieval-Augmented Generation (RAG), donde la calidad del resultado depende tanto de la capacidad generativa del modelo como de la recuperación contextual de información corporativa.

   En el contexto de este proyecto, el análisis se centra exclusivamente en hubs de integración de IA orientados a entornos empresariales, descartando deliberadamente plataformas low-code, orquestadores visuales o herramientas de automatización genérica, con el fin de mantener control arquitectónico, trazabilidad y alineación con atributos de calidad como seguridad, fiabilidad y escalabilidad.

   Las soluciones evaluadas se caracterizan por:

   - Exponer APIs estandarizadas para invocar múltiples modelos de IA.
   - Ofrecer capacidades nativas o integrables de RAG.
   - Incorporar mecanismos de gobernanza, seguridad y cumplimiento normativo.
   - Reducir el riesgo de *vendor lock-in* mediante abstracción del proveedor de modelos.

3.4.0.1. Hubs de IA en la nube (Enterprise)

Los hubs de inteligencia artificial en la nube (Enterprise AI Hubs) corresponden a plataformas administradas que centralizan el acceso a modelos fundacionales de distintos proveedores, ofreciendo capacidades empresariales de seguridad, gobernanza, auditoría y escalabilidad. Estas soluciones surgen como respuesta a la necesidad de las organizaciones de adoptar inteligencia artificial generativa sin incurrir en dependencia tecnológica de un único proveedor, manteniendo control sobre el uso de datos sensibles y los costos operativos (Gartner, 2024; Menlo Ventures, 2024).

Desde una perspectiva arquitectónica, los Enterprise AI Hubs funcionan como una capa de abstracción —frecuentemente denominada AI Gateway o LLM Gateway— que desacopla a las aplicaciones de negocio de los modelos de lenguaje subyacentes. Este enfoque se alinea con principios de diseño de arquitectura de software orientados al desacoplamiento, la interoperabilidad y la evolución independiente de componentes, permitiendo integrar múltiples modelos de diferentes fabricantes mediante APIs estandarizadas (Richards and Ford, 2020).

Plataformas empresariales como Amazon Bedrock, Azure OpenAI Service y Google Vertex AI ejemplifican este enfoque al ofrecer acceso centralizado a modelos fundacionales propios y de terceros, junto con mecanismos nativos de autenticación, control de acceso, registro de auditoría y cumplimiento normativo. Estas plataformas están diseñadas para el uso empresarial de modelos de lenguaje a gran escala y su integración segura en aplicaciones corporativas (Amazon Web Services, 2025; Microsoft, 2024; Google Cloud, 2024).

Asimismo, la literatura reciente destaca que los hubs de IA en la nube facilitan la implementación de arquitecturas tipo Retrieval-Augmented Generation (RAG), al integrarse con servicios de almacenamiento documental, motores de búsqueda semántica y bases de datos vectoriales. Esta integración permite enriquecer las respuestas generadas por los modelos con información corporativa actualizada y verificable, reduciendo el riesgo de alucinaciones y mejorando la fiabilidad del contenido generado (Lewis et al., 2020; Zhao et al., 2024).

Finalmente, los Enterprise AI Hubs habilitan estrategias avanzadas de gobierno del uso de IA, incluyendo el monitoreo de consumo, la trazabilidad de solicitudes y la aplicación de políticas de enrutamiento dinámico de modelos (AI Model Routing). Estas capacidades permiten optimizar el equilibrio entre calidad, latencia y costo, y son una característica distintiva de las plataformas empresariales de IA generativa en la nube (Amazon Web Services, 2025; Gartner, 2024).

Ejemplos: AWS Bedrock, Azure OpenAI + AI Search, Google Vertex AI, IBM watsonx.ai, Oracle AI Services, Anthropic Claude Enterprise.

Enfoque: Plataformas administradas que centralizan modelos de IA de distintos proveedores, con capacidades nativas de RAG y funciones avanzadas de seguridad, gobernanza y cumplimiento normativo.

En la Tabla 3.2 se presenta una comparación de los principales *Hubs de IA empresariales* clasificados por fabricante, con énfasis en las capacidades de compatibilidad e integración con distintos modelos y proveedores de inteligencia artificial. Esta comparación permite identificar el grado de apertura de cada solución y su alineación con enfoques multivendor. Por su parte, la Tabla 3.3 complementa el análisis anterior detallando otros aspectos relevantes, tales como características de gobernanza,

control de costos, seguridad, trazabilidad, soporte para arquitecturas RAG y facilidad de integración en entornos empresariales. En conjunto, ambas tablas proporcionan una visión integral para evaluar las alternativas de Hubs de IA desde una perspectiva técnica y arquitectónica.

Tabla 3.2: Comparación de hubs de IA empresariales multivendor

Hub de IA	**Compatibilidad RAG**	**Modelos multivendor compatibles**	**Integración empresarial**
Amazon Bedrock	Nativa (Knowledge Bases, OpenSearch)	Amazon, Anthropic, Meta, Mistral, Cohere	SAP, Salesforce, ServiceNow, APIs REST
Azure OpenAI / AI Studio	Integrada (Azure AI Search, Fabric)	OpenAI, Meta, Mistral, Cohere	Microsoft 365, SAP, Salesforce
Google Vertex AI	Nativa (Vertex AI RAG Engine)	Google, Meta, Anthropic, Mistral	Google Workspace, APIs REST
IBM watsonx.ai	Integrada (watsonx.data, vector DB)	IBM, Meta, Mistral	SAP, sistemas legacy, APIs
Oracle OCI Generative AI	Integrada (OCI Search, Vector DB)	Oracle, Meta, Cohere	Oracle ERP, APIs REST

Tabla 3.3: Comparación de hubs de IA empresariales

Hub	**RAG**	**Multivendor**	**Integración / APIs**	**Fortalezas**	**Limitaciones**	**Casos de uso**
Amazon Bedrock	Nativa	Sí	Alta	Gobierno, escalabilidad	Costos, lock-in	Copilotos, RAG
Azure OpenAI	Integrada	Sí	Alta	Seguridad, trazabilidad	Lock-in, costos	Asistentes
Google Vertex AI	Nativa	Sí	Media–Alta	Resiliencia	Costos	Búsqueda
IBM watsonx.ai	Integrada	Sí	Media	Compliance	Ecosistema cerrado	Asistentes
Oracle OCI GenAI	Integrada	Parcial	Alta (Oracle)	Integración ERP	Menos modelos	Automatización
SAP AI / Joule	Integrada	Parcial	Alta (SAP)	Procesos negocio	Dependencia SAP	Ofertas
Salesforce Einstein	Integrada	Parcial	Alta (CRM)	Contexto comercial	Limitado a CRM	Ventas
ServiceNow Now Assist	Integrada	Parcial	Alta (ITSM)	Automatización IT	Enfoque específico	Soporte

3.4.0.2. Bases vectoriales y recuperación contextual

Ejemplos: Pinecone, Weaviate, Qdrant, Milvus, FAISS, Redis Vector, Elasticsearch Vector, pgvector.

Enfoque: Motores de búsqueda semántica y almacenamiento vectorial utilizados como núcleo de la capa de recuperación en sistemas RAG.

Modelos compatibles (para embeddings): OpenAI Embeddings, Cohere Embeddings, Titan Embeddings, E5, BGE, InstructorXL, Jina, GTE.

Compatibilidad RAG: Nativa, habilitan la búsqueda y recuperación de fragmentos contextuales relevantes para el grounding del modelo.

Fortalezas:

- Alto rendimiento y precisión en búsquedas semánticas.
- Soporte para metadatos, filtros RBAC y búsquedas híbridas.
- Escalabilidad horizontal y despliegue flexible (cloud u on-premise).

Limitaciones: No gestionan la generación ni la evaluación; requieren infraestructura adicional.

Casos de uso: Indexación semántica, búsqueda de conocimiento empresarial, recuperación contextual para sistemas RAG.

3.4.1. Análisis de requerimientos de Infraestructura Virtual SAS

Esta sección presenta el análisis de requerimientos funcionales y no funcionales de la empresa Infraestructura Virtual SAS, orientado a identificar las necesidades específicas del proceso de preventa y generación de ofertas comerciales. El objetivo es establecer las condiciones técnicas, operativas y de información que debe cumplir la solución arquitectónica propuesta, considerando los flujos actuales del proceso, las limitaciones identificadas y las oportunidades de automatización. A partir de este análisis, se definen los insumos que guiarán el diseño de la arquitectura.

3.4.1.1. Flujo de trabajo actual

El proceso actual de generación de ofertas comerciales en Infraestructura Virtual SAS está regulado por el procedimiento DS-PRO-02 “Flujograma del Proceso de Preventa”.

Tabla 3.4: Flujo de trabajo preventa

Etapa	Descripción de la Actividad	Responsable	Entradas / Salidas
1. Registro y categorización	El Director de Preventa recibe la oportunidad en Odoo y asigna preventa según capacidad, prioridad y especialidad.	Director de Preventa	Entrada: Oportunidad comercial (Odoo CRM) Salida: Asignación del preventa (Odoo) / Notificación (Teams o correo).
2. Reunión de entendimiento	Reunión con el cliente para recopilar requerimientos técnicos y comerciales.	Account Manager / Preventa	Entrada: Solicitud del cliente (correo / Teams / reunión) Salida: Levantamiento de información consultiva (Word / notas Teams).
3. Análisis técnico y diseño	El preventa elabora la arquitectura técnica y económica preliminar; puede solicitar apoyo a especialistas.	Preventa de Soluciones	Entrada: Información del cliente (Word / correo / Odoo) Salida: Propuesta técnica preliminar (Word) / Lista de requerimientos para cotización (Excel/Word).
4. Gestión de compras	Solicitud de cotizaciones o precios actualizados a Compras (mínimo 3 días hábiles).	Preventa / Compras	Entrada: Requerimiento de cotización (correo / Excel) Salida: Cotización del proveedor (PDF / correo electrónico).
5. Costeo y verificación	El preventa elabora el cuadro de costos y lo valida con el Account Manager y Director de Preventa.	Preventa / Dirección	Entrada: Cotizaciones (PDF/correo) Salida: Cuadro de costos validado (Excel) / Aprobación (correo / Teams).
6. Evaluación de riesgos	Se analizan los posibles riesgos técnicos, financieros y operativos documentándolos en la Matriz de Riesgos.	Preventa de Soluciones	Entrada: Datos del proyecto (Word/Excel/Odoo) Salida: Matriz de riesgos (Excel).

Etapa	Descripción de la Actividad	Responsable	Entradas / Salidas
7. Revisión y ajustes	Se realizan modificaciones según el tipo de cambio: económico o técnico.	Account Manager / Preventa	Entrada: Solicitud de cambio (correo / Teams / Odoo) Salida: Oferta ajustada (Word / Excel actualizados).
8. Entrega final	Entrega de documentos completos al área comercial para presentación al cliente.	Preventa / Comercial	Entrada: Documentos validados (Word, Excel, PDFs) Salida: Propuesta final (Word/PDF) / Envío al cliente (correo) / Registro en Odoo.

3.4.1.2. Requerimientos no funcionales

Durante el análisis se identificaron requerimientos no funcionales implícitos en el proceso actual que deben ser garantizados por la futura arquitectura:

Categoría	Requerimiento actual / esperado	Justificación
Disponibilidad	Acceso al flujo y documentación desde cualquier ubicación.	El preventa trabaja con múltiples cuentas y clientes.
Rendimiento	Reducción del tiempo de respuesta por propuesta (¡2 días).	Actualmente el promedio supera los 5 días hábiles.
Fiabilidad	Control de versiones y trazabilidad entre revisiones.	Se detectan inconsistencias en documentos finales.
Seguridad	Control de acceso por rol (preventa, comercial, dirección).	Protección de información confidencial.
Usabilidad	Interfaces unificadas para acceder a información y plantillas.	Hoy las fuentes están dispersas (Excel, Odoo, correos).
Escalabilidad	Soporte para múltiples propuestas simultáneas sin degradación.	El volumen crece a más de 12 oportunidades semanales.

Alcance de los requerimientos no funcionales. Los requerimientos no funcionales identificados fueron analizados con base en los objetivos del proyecto. En este sentido, los atributos de rendimiento, fiabilidad, seguridad y escalabilidad se consideran prioritarios y forman parte explícita de la evaluación arquitectónica, conforme a los objetivos específicos definidos.

El atributo de usabilidad, aunque relevante en sistemas orientados al usuario final, no se incluye dentro del alcance de evaluación del presente trabajo, dado que el enfoque del proyecto se centra en el diseño

arquitectónico y en atributos de calidad críticos desde una perspectiva técnica. No obstante, la arquitectura propuesta no limita su evaluación futura en fases posteriores de implementación o mejora del sistema.

3.4.1.3. Análisis diagnóstico de puntos de bloqueo, limitaciones y oportunidades de mejora

Esta sección consolida los principales puntos de bloqueo identificados en el proceso actual de generación de ofertas comerciales, junto con las limitaciones estructurales del flujo de trabajo y las oportunidades de mejora. El análisis se fundamenta en la observación del proceso vigente y entrevistas con el equipo de preventa, estableciendo métricas asociadas al nivel de completitud y precisión (*accuracy*) de las propuestas.

Puntos de bloqueo identificados. La Tabla 3.6 resume los bloqueos principales, su impacto y la causa raíz asociada.

Tabla 3.6: Principales puntos de bloqueo identificados

Bloqueo identificado	**Impacto en el proceso**	**Causa raíz**
Dependencia del conocimiento individual	Riesgo alto de error o retraso cuando un preventa está ausente.	Falta de documentación estructurada y repositorio central.
Retrasos en cotizaciones	Aplazan la entrega hasta 3 días o más.	Tiempos mínimos de respuesta de proveedores externos.
Duplicidad de información	Reprocesos y versiones inconsistentes.	No existe un sistema centralizado de control de versiones.
Validaciones manuales de costos	Errores humanos y demoras en la revisión.	Falta de automatización de checklists y control de márgenes.
Reutilización limitada de información	Reconstrucción de propuestas similares desde cero.	Ausencia de mecanismos de búsqueda o plantillas inteligentes.
Escasa trazabilidad	Dificultad para rastrear cambios entre versiones.	Control documental manual mediante archivos locales.

Limitaciones actuales del proceso. Con base en lo observado, se identifican las siguientes limitaciones estructurales:

- No existe integración entre Odoo, hojas de cálculo y documentos técnicos.
- Ausencia de repositorio de conocimiento histórico: los documentos se almacenan en carpetas sin estructura semántica.
- Falta de indicadores de desempeño (KPIs): el área de preventa no mide tiempos de elaboración, revisiones o causas de rechazo.
- Carencia de automatización documental: las propuestas se redactan y formatean manualmente.

- Dificultad de colaboración simultánea: múltiples usuarios editan los mismos documentos sin control de versiones.

Oportunidades de mejora. La Tabla 3.7 presenta oportunidades alineadas con la automatización, la trazabilidad y la eficiencia del proceso.

Tabla 3.7: Oportunidades de mejora identificadas

Área	**Propuesta de mejora**	**Resultado esperado**
Gestión del conocimiento	Implementar un repositorio central con motor de búsqueda semántico (IA tipo RAG).	Reutilización de propuestas históricas y reducción de tiempos.
Automatización documental	Integrar un módulo que genere documentos técnicos y económicos automáticos.	Estandarización de formatos y reducción de errores.
Integración de sistemas	Sincronizar Odoo, gestor documental y cotizaciones mediante APIs.	Flujo continuo de información entre áreas.
Validación de costos	Automatizar reglas de validación financiera (márgenes, rentabilidad).	Control eficiente de precios y minimización de reprocesos.
Seguimiento de rendimiento	Definir métricas de tiempo y cumplimiento en preventa.	Control de eficiencia y priorización de mejoras.
Trazabilidad y control de versiones	Implementar un gestor documental con control de cambios y permisos.	Mayor seguridad y consistencia en la información.

3.4.2. Drivers de arquitectura

Los drivers de arquitectura representan los factores clave que influyen en las decisiones de diseño de la solución propuesta para la automatización del proceso de generación de ofertas comerciales. Estos drivers se derivan directamente del análisis diagnóstico del proceso actual, de las limitaciones identificadas y de los objetivos de mejora definidos, y permiten establecer criterios claros para la selección de patrones, tecnologías y componentes arquitectónicos.

En este contexto, los drivers se agrupan principalmente en dimensiones de calidad, alineadas con el modelo de calidad del software definido por la norma ISO/IEC 25010, considerando además aspectos de escalabilidad, interoperabilidad y sostenibilidad operativa propios de entornos empresariales modernos. La Tabla 3.8 presenta los drivers de arquitectura identificados y su correspondencia con las dimensiones de calidad relevantes.

Tabla 3.8: Drivers de arquitectura

Tipo de driver	**Descripción (en el contexto de la automatización de ofertas comerciales)**	**Dimensión ISO/IEC 25010 asociada**
Rendimiento y eficiencia	La arquitectura debe permitir generar una propuesta comercial completa en menos de 2 días hábiles, reduciendo los tiempos de procesamiento y respuesta.	Performance efficiency
Fiabilidad	El sistema debe asegurar la consistencia de los datos y la disponibilidad continua del flujo de trabajo, incluso ante fallos de componentes.	Reliability
Usabilidad	Las interfaces deben ser intuitivas y adaptadas al perfil técnico-comercial de los usuarios (preventa, comercial y dirección).	Usability
Seguridad	Debe implementarse control de acceso por roles, cifrado de la información en tránsito y en reposo, así como trazabilidad completa de las modificaciones realizadas sobre los documentos.	Security
Mantenibilidad	La arquitectura debe ser modular y desacoplada, permitiendo la sustitución o actualización de componentes (por ejemplo, modelos de IA) sin afectar al resto del sistema.	Maintainability
Portabilidad	La solución debe poder desplegarse en diferentes entornos cloud (por ejemplo, AWS o Azure) sin requerir cambios sustanciales en el código o la infraestructura.	Portability
Compatibilidad e interoperabilidad	Debe integrarse de forma transparente con los sistemas existentes (Odoo, SharePoint, correo electrónico y gestor documental) mediante APIs REST o servicios en la nube.	Compatibility
Escalabilidad y capacidad de expansión	La arquitectura debe soportar el incremento de oportunidades comerciales y usuarios concurrentes sin una degradación significativa del rendimiento.	Performance / Scalability
Sostenibilidad técnica y operativa	Se prioriza el uso de servicios cloud administrados para reducir el esfuerzo de mantenimiento, el consumo de recursos y los costos operativos.	Sustainability

Nota: Los drivers arquitectónicos se definieron conforme al modelo de calidad del software ISO/IEC 25010, considerando los atributos relevantes para el diseño.

Capítulo 4

Diseño

4.1. Propósito y alcance del diseño arquitectónico

Este capítulo presenta el diseño arquitectónico de la solución propuesta para la automatización de ofertas comerciales en Infraestructura Virtual SAS. El propósito es describir la arquitectura desde una perspectiva estructural y de decisión, detallando los elementos observables del sistema, sus límites, los principales contenedores ejecutables, los componentes internos relevantes y las decisiones que soportan la integración de inteligencia artificial generativa bajo un enfoque multivendor.

El diseño se documenta utilizando el modelo C4 como mecanismo de representación, complementado con la definición de componentes tecnológicos, patrones arquitectónicos adoptados y la especificación del componente *Hub de IA* con su estrategia de enrutamiento dinámico de modelos (*AI Model Routing*). Lo anterior asegura trazabilidad entre el diagnóstico del capítulo anterior, los drivers de arquitectura y las decisiones técnicas materializadas en la propuesta.

4.2. Diagrama de Contexto (Modelo C4)

Con el fin de contextualizar la arquitectura propuesta, esta sección presenta una vista de alto nivel del sistema de automatización de ofertas comerciales dentro del entorno empresarial objeto de estudio. El diagrama de contexto identifica a los actores humanos y sistemas externos que interactúan con la solución, así como los principales flujos de información que habilitan la creación, validación y entrega de propuestas comerciales.

El flujo inicia con la gestión de oportunidades comerciales por parte del *Account Manager*, quien registra y actualiza la información relevante en el CRM. A partir de dichas oportunidades, el equipo de preventa define alcances técnicos, riesgos y costos asociados. Esta información es consumida por el sistema de Automatización de Ofertas Comerciales, el cual genera propuestas preliminares apoyándose en capacidades de inteligencia artificial generativa y recuperación aumentada por conocimiento (RAG). Para ello, el sistema accede a repositorios documentales con plantillas, anexos y ofertas previas, favoreciendo consistencia y reutilización del conocimiento organizacional.

Durante el proceso, la propuesta generada pasa por validaciones y aprobaciones del equipo de preventa y dirección. Complementariamente, el sistema puede interactuar con portales de fabricantes/proveedores para obtener cotizaciones y descuentos, así como con fuentes internas de costos y precios, garantizando que la información económica utilizada se encuentre actualizada. Finalmente, una vez validada, la propuesta se distribuye mediante canales corporativos de comunicación integrados con el servicio de correo empresarial.

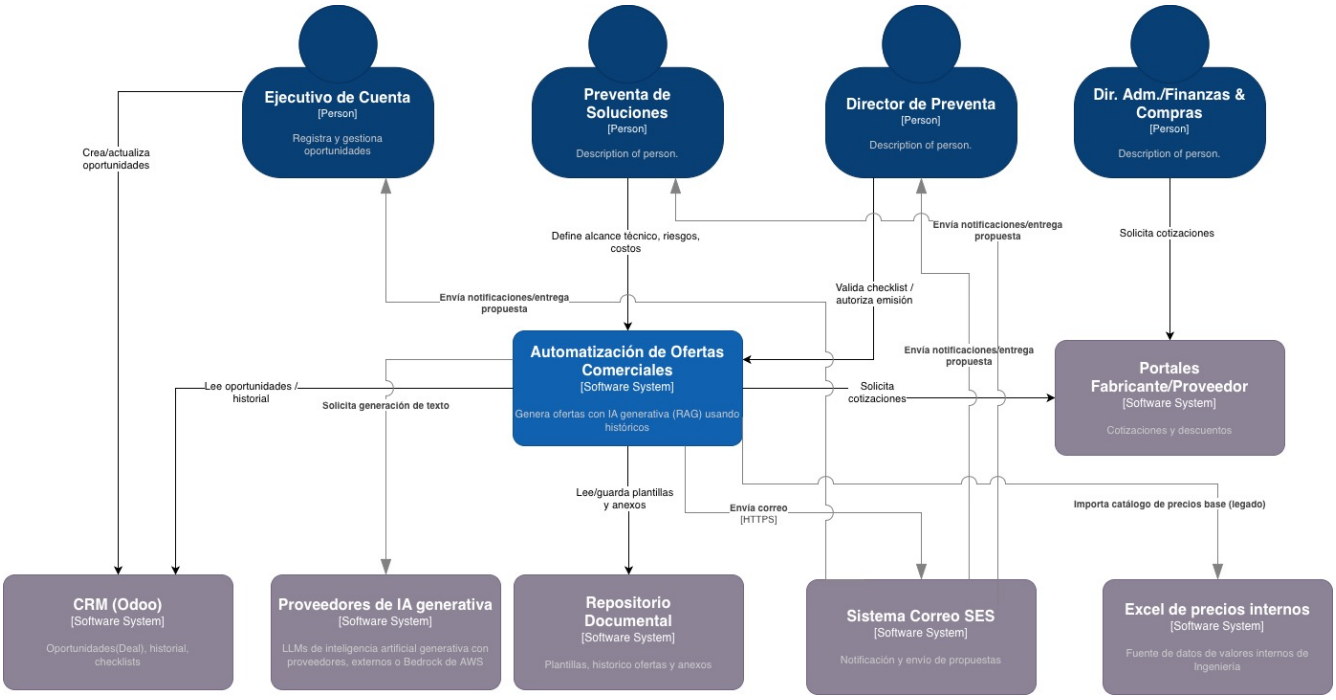


Figura 4.1: Diagrama de Contexto (C4)

4.3. Diagrama de Contenedores (Modelo C4)

El Diagrama de Contenedores, basado en el modelo C4, describe la arquitectura del sistema identificando los contenedores ejecutables, almacenes de datos y dependencias externas con las que interactúa la solución. Este nivel permite comprender cómo se distribuyen responsabilidades, cómo fluye la información entre módulos y qué integraciones soportan el proceso de automatización.

En el límite del sistema se ubica el **Portal Web de Preventa**, encargado de ofrecer la interfaz para gestionar oportunidades, solicitar generación de propuestas, revisar versiones y descargar documentos finales. El acceso se realiza mediante HTTPS y requiere autenticación previa. La autenticación y autorización se gestiona mediante el contenedor de **Autenticación**, el cual implementa control de acceso basado en roles (RBAC) y emisión de tokens mediante estándares como OAuth 2.0.

El **API Backend** constituye el núcleo de orquestación, exponiendo servicios consumidos por el Portal Web, aplicando reglas de negocio, gestionando seguridad y trazabilidad, e integrándose con el CRM corporativo para consultar información de oportunidades e historial. El contenedor de **Generación de Propuestas** coordina el flujo de generación, solicitando contexto al motor de recuperación, invocando capacidades de IA para producir contenido, gestionando estados del proceso y controlando el versionado de las propuestas.

La solución incorpora una **Base de Datos Vectorial** para almacenar embeddings y metadatos de documentos históricos, habilitando búsquedas semánticas durante la generación. El **Repositorio Documental** centraliza plantillas y anexos, actuando como fuente principal de conocimiento. Adicionalmente, una **Base de Datos Transaccional** almacena oportunidades, estados del flujo, versiones, auditoría y

trazabilidad.

El **Hub de IA Generativa** centraliza la interacción con modelos de IA, desacoplando la lógica de negocio de proveedores específicos y permitiendo invocar modelos gestionados (p. ej., Amazon Bedrock) y modelos externos vía APIs. Este componente aplica políticas de seguridad, trazabilidad y enrutamiento dinámico según el tipo de tarea y condiciones definidas.

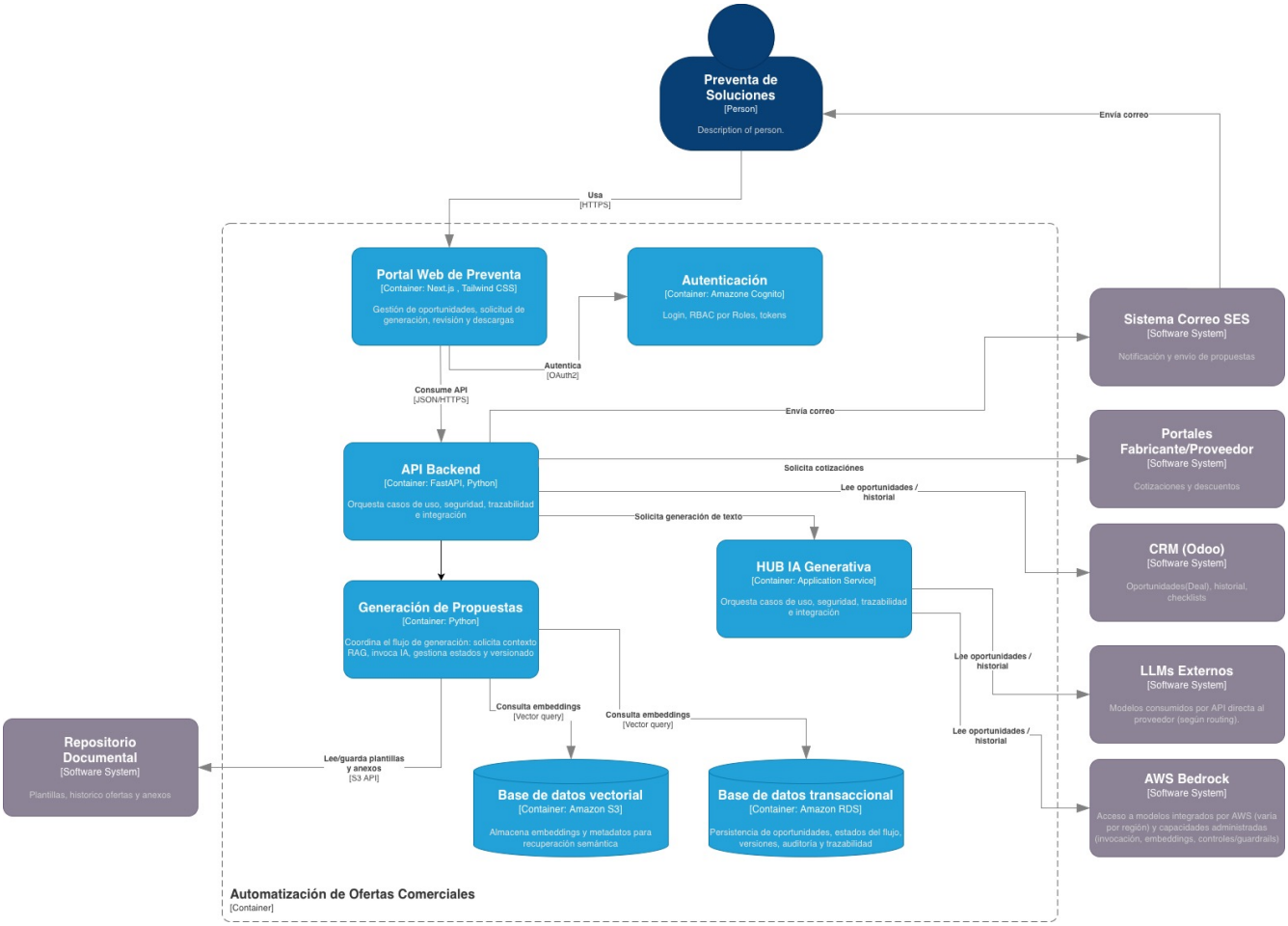


Figura 4.2: Diagrama de Contenedores (C4)

4.4. Diagrama de Componentes (Modelo C4)

El Diagrama de Componentes describe la estructura interna de los principales contenedores de la solución de Automatización de Ofertas Comerciales, detallando los componentes software que soportan el flujo de negocio y sus responsabilidades específicas, así como sus interacciones con sistemas externos.

La arquitectura se centra en tres contenedores principales: *API Backend*, *Servicio de Generación de Propuestas* y *Hub de IA Generativa*, dentro del cual se integra el motor RAG. Adicionalmente, el diagrama incluye contenedores de soporte como el *Portal Web de Preventa* y el servicio de *Autenticación*, que actúan

como puntos de entrada y control de acceso a la solución.

4.4.1. Componentes del API Backend

El *API Backend* constituye el punto central de orquestación de la solución y concentra los componentes responsables de exponer los servicios del sistema, aplicar reglas de negocio y coordinar la interacción entre los distintos módulos internos y externos.

Los componentes principales que conforman este contenedor son:

- **API Controllers**: exponen los endpoints REST consumidos por el Portal Web de Preventa para la gestión de oportunidades, la solicitud de generación de propuestas, la consulta de estados y la descarga de versiones, utilizando comunicación JSON sobre HTTPS.
- **Middleware de Autorización (OAuth2/JWT)**: valida los tokens emitidos por el proveedor de identidad externo y aplica control de acceso basado en roles (RBAC), garantizando que cada usuario acceda únicamente a las funcionalidades permitidas.
- **Orquestador de Casos de Uso**: coordina los flujos principales del negocio de preventa, invocando los distintos servicios y adaptadores necesarios para la creación, revisión y publicación de propuestas, desacoplando la lógica de presentación de la lógica de dominio.
- **Gestor de Estados y Versiones**: administra el ciclo de vida de las propuestas (borrador, en revisión, publicadas), manteniendo el historial de versiones, los estados del proceso y la información de auditoría asociada.
- **Adaptador de Integración con CRM (Odoo)**: encapsula la comunicación con el sistema CRM para la consulta de oportunidades, historial comercial y checklist, evitando dependencias directas del dominio hacia sistemas externos.
- **Adaptador de Cotizaciones**: gestiona la integración con portales de fabricantes para la solicitud y recepción de cotizaciones y descuentos, actuando como intermediario entre el backend y los sistemas externos de proveedores.
- **Adaptador de Notificaciones**: gestiona el envío de correos electrónicos y alertas mediante el servicio de mensajería (SES), notificando a los distintos actores sobre eventos relevantes del proceso de generación de propuestas.

4.4.2. Componentes del Servicio de Generación de Propuestas

El contenedor de *Generación de Propuestas* implementa la lógica específica de automatización documental y la coordinación del proceso de generación asistida por inteligencia artificial. Este servicio actúa como intermediario entre el backend de negocio y el Hub de IA Generativa.

Los componentes relevantes de este contenedor son:

- **Validador de Entradas**: verifica la completitud, coherencia y consistencia mínima de los datos de la oportunidad antes de iniciar el proceso de generación de la propuesta.

- **Orquestador de Generación**: coordina las distintas etapas del flujo de generación, incluyendo la validación de datos, la solicitud de contexto al Hub de IA, la generación de contenido, la revisión y la publicación de versiones.

- **Lector de Prompts**: gestiona la lectura y estructuración de los prompts utilizados en la generación, definiendo la organización del contenido y las secciones de la propuesta.

- **Ensamblador Documental**: consolida el contenido generado por la IA con plantillas corporativas, textos base y anexos, produciendo los documentos finales de la propuesta.

- **Gestor de Artefactos**: administra el almacenamiento de borradores y versiones finales de las propuestas, persistiendo los artefactos generados y sus metadatos asociados en el repositorio documental externo.

4.4.3. Componentes del Hub de IA Generativa (incluyendo Motor RAG)

El *Hub de IA Generativa* actúa como una capa de abstracción entre la aplicación y los proveedores de modelos de lenguaje, centralizando el uso de inteligencia artificial y desacoplando la arquitectura de proveedores específicos. En este contenedor se integra el motor RAG como parte fundamental del proceso de generación contextual.

Los componentes principales del Hub de IA son:

- **Políticas y Guardrails**: aplica controles de seguridad, cumplimiento normativo y validación de parámetros antes de invocar los modelos de lenguaje, garantizando un uso controlado de la IA.

- **Motor RAG**: realiza la recuperación semántica de información relevante desde la base de datos vectorial, construyendo el contexto necesario para enriquecer la generación de contenido mediante modelos de lenguaje.

- **AI Model Router**: selecciona dinámicamente el modelo de lenguaje más adecuado según el tipo de tarea, el costo, la latencia y los criterios definidos por la política de uso.

- **Adaptadores de Proveedores de IA**: encapsulan la integración con modelos administrados (Amazon Bedrock) y modelos externos consumidos vía API directa, evitando dependencias directas del dominio hacia proveedores específicos.

- **Gestor de Resiliencia**: implementa mecanismos de tolerancia a fallos, como reintentos controlados, tiempos de espera y estrategias de conmutación entre proveedores de IA.

- **Registro y Trazabilidad**: almacena información de auditoría relacionada con la ejecución de los modelos, incluyendo prompts, contexto utilizado, métricas y tiempos de respuesta, persistiendo estos datos en la base transaccional.

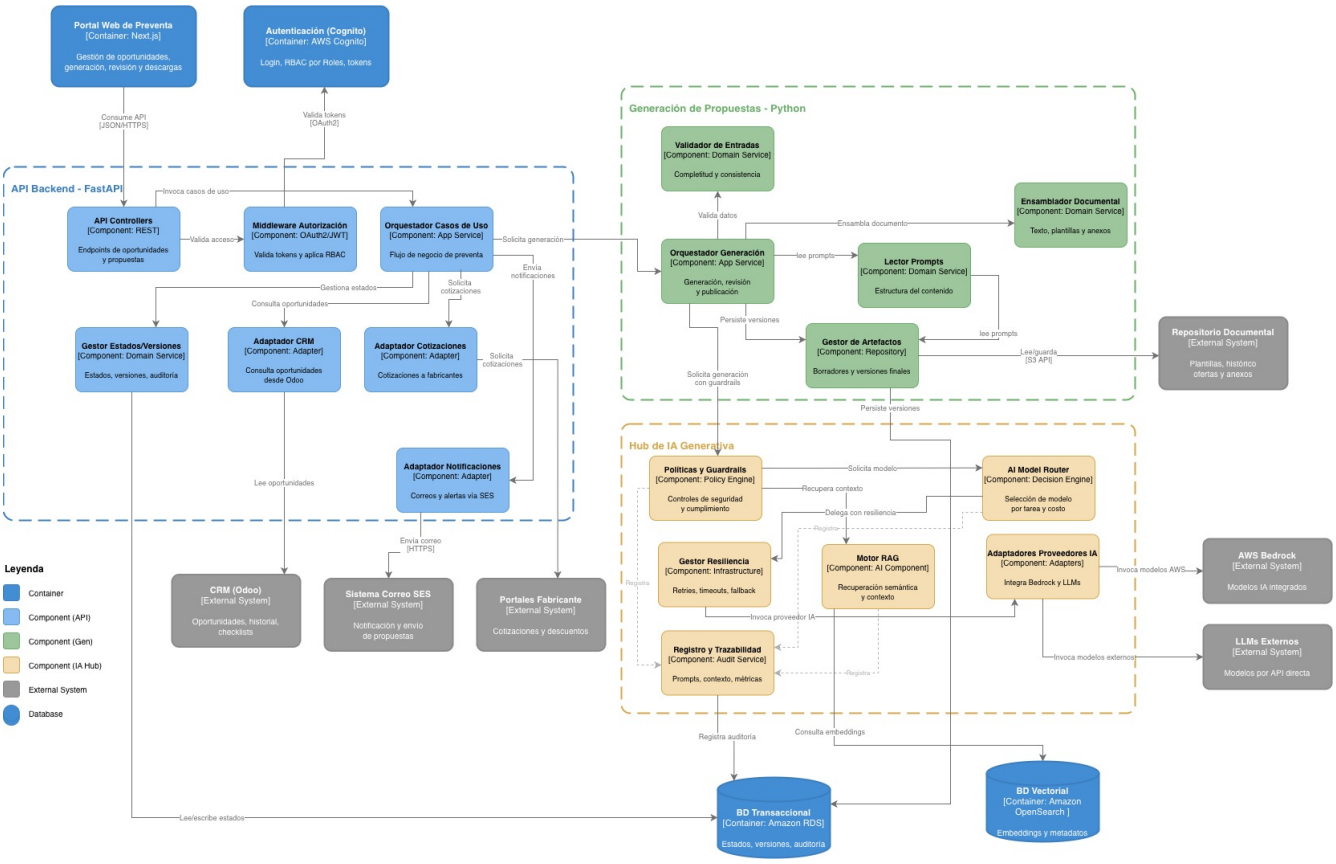


Figura 4.3: Diagrama de Componentes (C4)

4.5. Componentes tecnológicos de la solución

Esta sección describe los componentes tecnológicos seleccionados para implementar la arquitectura en un entorno *cloud*, manteniendo un enfoque agnóstico al proveedor (*multi-cloud*). En lugar de depender de una sola plataforma, se identifican capacidades equivalentes presentes en los proveedores con mayor adopción empresarial —**Amazon Web Services (AWS), Microsoft Azure y Google Cloud Platform (GCP)**— como una decisión metodológica basada en criterios de adopción, madurez tecnológica y cobertura funcional.

Estas plataformas concentran la mayor presencia en entornos corporativos, ofrecen disponibilidad multi-región y disponen de servicios administrados en cómputo, almacenamiento, bases de datos, seguridad, observabilidad e inteligencia artificial. Por esta razón, se consideran referentes adecuados para evaluar la viabilidad de una arquitectura moderna y escalable.

Si bien existen otras alternativas, como nubes regionales o soluciones auto-gestionadas, estas suelen presentar limitaciones en términos de alcance global, ecosistema de herramientas, soporte empresarial o integración con servicios avanzados de IA.

Tabla 4.1: Equivalencia de componentes cloud y su función en la arquitectura propuesta

Capa	**Capacidad**	**AWS**	**Azure**	**GCP**	**Función en la arquitectura propuesta**
Red y cómputo	Red virtual	VPC	VNet	VPC	Segmentación en subredes públicas y privadas para aislar backend, base de datos y servicios de IA bajo principios de mínimo privilegio.
	Contenedores administrados	ECS Fargate / EKS	AKS / Container Apps	GKE / Cloud Run	Ejecución del backend y del Hub de IA con escalado automático y despliegue continuo.
	Funciones serverless	Lambda	Azure Functions	Cloud Functions	Procesamiento asíncrono (indexación de documentos, generación de embeddings, disparadores RAG).
	Balanceador de carga	ALB	Application Gateway	Cloud Load Balancing	Exposición segura de APIs y distribución de tráfico hacia servicios internos.
Datos y RAG	Almacenamiento de objetos	S3	Blob Storage	Cloud Storage	Repositorio central de plantillas y propuestas históricas que alimentan el flujo RAG.
	Base de datos relacional	RDS	Azure Database for PostgreSQL	Cloud SQL	Persistencia transaccional (oportunidades, estados, auditoría y versionado).
	Vector Store / Búsqueda	OpenSearch / Aurora (vector)	Azure AI Search / Cosmos DB (vector)	Vertex AI Vector Search	Indexación semántica para recuperación contextual dentro del motor RAG.

Capa	Capacidad	AWS	Azure	GCP	Función en la arquitectura propuesta
IA / Hub	Modelos fundacionales	Bedrock	Azure OpenAI	Vertex AI	Generación de contenido y análisis semántico mediante LLMs con controles empresariales.
	Embeddings	Bedrock Embeddings	Azure OpenAI Embeddings	Vertex AI Embeddings	Conversión de documentos en representaciones vectoriales para búsqueda por similitud.
	Motor RAG	Orquestado sobre Lambda/ECS	Orquestado sobre Functions/AKS	Orquestado sobre Cloud Run/GKE	Construcción de contexto combinando recuperación semántica y generación contextualizada.
Integración	Gestión de APIs	API Gateway	API Management	API Gateway / Apigee	Exposición controlada de servicios con autenticación, autorización y control de tráfico.
	Mensajería / eventos	SQS / EventBridge	Service Bus / Event Grid	Pub/Sub	Desacoplamiento de procesos asíncronos y comunicación basada en eventos.
Seguridad	Identidad y control de acceso	IAM	Entra ID	IAM	Gestión de identidades, roles y políticas de acceso bajo principio de mínimo privilegio.
	Cifrado y gestión de secretos	KMS / Secrets Manager	Key Vault	KMS / Secret Manager	Protección de datos en tránsito y reposo, y almacenamiento seguro de credenciales.
Observabilidad	Monitoreo y métricas	CloudWatch	Azure Monitor	Cloud Monitoring	Supervisión de rendimiento, métricas operativas y alertas.
	Auditoría y trazabilidad	CloudTrail / X-Ray	Activity Logs / App Insights	Audit Logs / Cloud Trace	Registro de eventos, trazabilidad de transacciones y soporte a cumplimiento normativo.

4.6. Patrones arquitectónicos aplicados en la solución

Esta sección lista los patrones arquitectónicos adoptados en la solución y su aplicación directa en el contexto del sistema propuesto:

- **Arquitectura en capas**: separa presentación (Portal Web), aplicación (API Backend) e infraestructura (datos, IA e integraciones) para mejorar mantenibilidad.
- **Servicios / microservicios**: descomposición por capacidades (generación, documentos, RAG, hub de IA), habilitando escalado selectivo y despliegues independientes.
- **Arquitectura dirigida por eventos**: procesos asíncronos (indexación, embeddings, notificaciones) para reducir acoplamiento y mejorar desempeño.
- **Orquestación de procesos (Saga/Workflow)**: coordinación del flujo de generación con manejo de estados, reintentos y compensaciones.
- **API Gateway / BFF**: centraliza autenticación, control de acceso y exposición consistente de endpoints hacia la UI.
- **Hexagonal (Ports & Adapters)**: encapsula integraciones con CRM, almacenamiento, mensajería y proveedores de IA, reduciendo *vendor lock-in*.
- **Resiliencia (Circuit Breaker, Retry, Bulkhead)**: continuidad operativa ante fallos o degradación de servicios externos, especialmente en IA.
- **RAG + AI Gateway**: recuperación contextual y orquestación de múltiples modelos con políticas de seguridad y trazabilidad.

4.7. Diseño del componente Hub de IA generativa

El *Hub de IA generativa* constituye el componente central de integración, control y orquestación dentro de la arquitectura propuesta. Su propósito principal es desacoplar la lógica de negocio del sistema respecto a los proveedores de modelos de inteligencia artificial, centralizando la gestión de solicitudes, la aplicación de políticas transversales y la implementación de la estrategia de enrutamiento dinámico de modelos.

Esta centralización no solo reduce el acoplamiento entre el sistema y servicios externos, sino que también permite introducir mecanismos uniformes de gobernanza, trazabilidad y control de costos, aspectos críticos en entornos empresariales que integran capacidades de IA generativa.

Como se muestra en la Figura 4.4, el Hub se organiza bajo una arquitectura en capas que permite separar responsabilidades, minimizar dependencias cruzadas y facilitar la evolución independiente de cada módulo. Esta estructuración favorece la incorporación de nuevos proveedores, la adaptación ante cambios tecnológicos y la aplicación homogénea de controles de seguridad y observabilidad en todo el flujo de generación.

La arquitectura del Hub se compone de cinco capas claramente diferenciadas:

- **Ingress Layer.** Responsable de la recepción, autenticación y validación inicial de las solicitudes provenientes del sistema consumidor. Incluye mecanismos de autenticación mediante proveedor de identidad, validación estructural de entradas (esquema y tamaño), registro de peticiones y control de tasa (*rate limiting*). Esta capa actúa como primera línea de defensa, asegurando que únicamente solicitudes válidas, autorizadas y correctamente formateadas ingresen al flujo interno del sistema, reduciendo riesgos de abuso o uso indebido.

- **Middleware Layer.** Encargada de aplicar políticas y transformaciones transversales antes de la invocación del modelo. Entre sus funciones se incluyen el filtrado de contenido (*guardrails*), transformación y estandarización de prompts, enriquecimiento contextual mediante el motor RAG y estimación preliminar de costos asociados a la inferencia. Esta capa desacopla la preparación lógica de la solicitud respecto a la ejecución del modelo, permitiendo que las reglas de negocio y control puedan evolucionar sin modificar los adaptadores externos.

- **Routing Layer.** Implementa la estrategia de selección dinámica de modelos descrita en la sección anterior. En esta capa se ejecuta el cálculo multi-criterio que evalúa desempeño esperado, costo, latencia y confiabilidad histórica de cada modelo disponible. Asimismo, incorpora mecanismos de balanceo de carga, verificación de salud (*health checks*), control de fallos mediante *circuit breaker* y estrategias de *fallback*. Esta capa constituye el núcleo decisional del Hub y concentra la lógica de optimización adaptativa del sistema.

- **Egress Layer.** Contiene los adaptadores específicos hacia cada proveedor de modelos (AWS Bedrock, OpenAI, Anthropic, Google Vertex AI o modelos autoalojados). El uso del patrón de adaptadores permite encapsular las particularidades de cada API externa, reduciendo la dependencia directa del resto del sistema respecto a cambios en contratos, versiones o políticas de los proveedores. Esta capa es fundamental para minimizar el riesgo de *vendor lock-in* y facilitar la sustitución o incorporación de nuevos modelos.

- **Observability Layer.** Proporciona monitoreo integral, trazabilidad de solicitudes y control de costos mediante recolección de métricas, trazas distribuidas y registro de auditoría. Permite analizar latencia, consumo de tokens, decisiones de enrutamiento y eventos de resiliencia. Esta capa soporta procesos de gobernanza, análisis de desempeño y mejora continua, garantizando transparencia y capacidad de supervisión sobre el comportamiento del sistema.

La disposición en capas responde a principios de separación de responsabilidades, bajo acoplamiento y alta cohesión. Esta organización facilita el escalado horizontal de componentes críticos, fortalece la resiliencia ante fallos parciales y habilita una evolución tecnológica controlada, particularmente relevante en un entorno caracterizado por la rápida evolución de modelos de inteligencia artificial generativa.

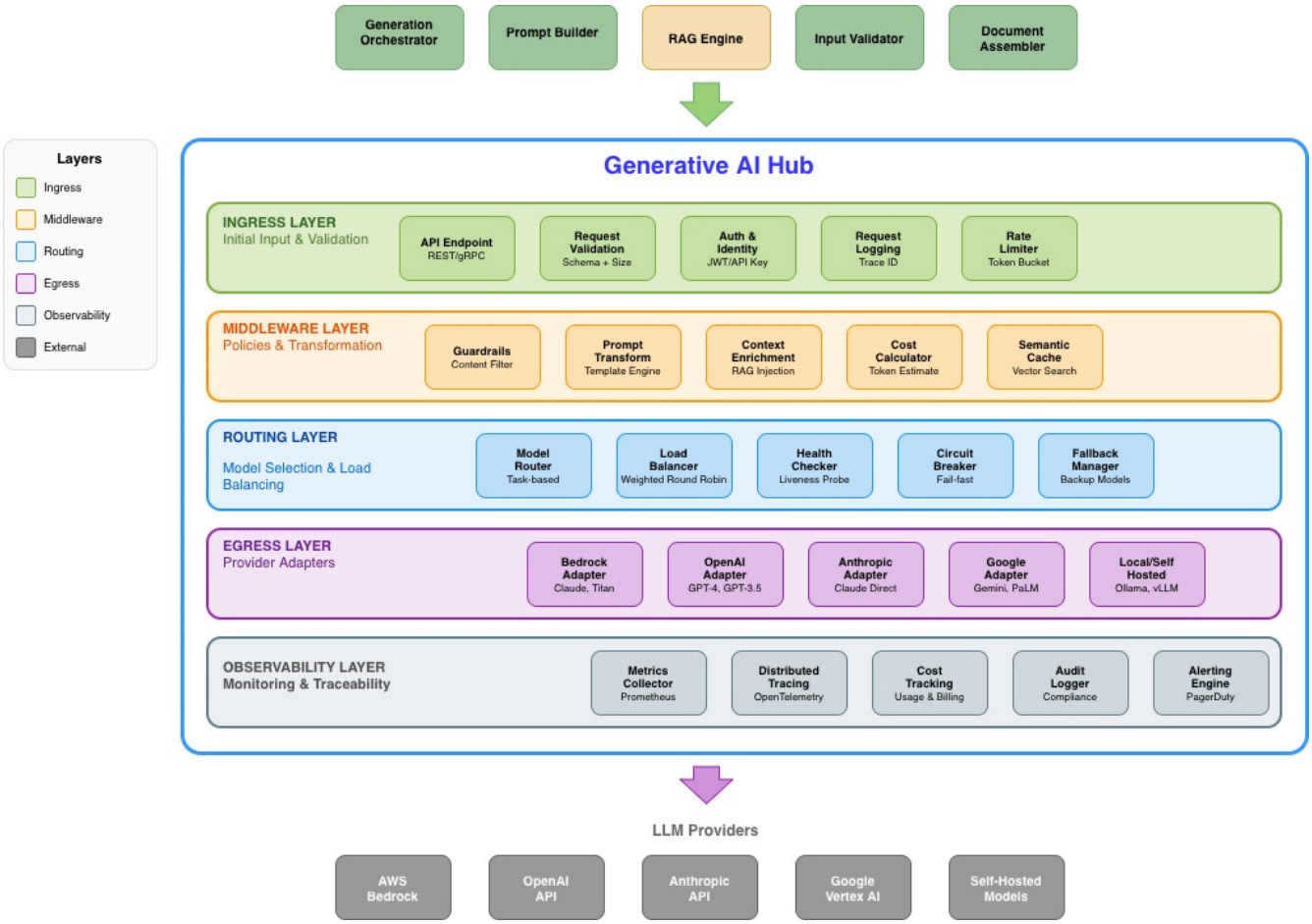


Figura 4.4: Arquitectura en capas HUB IA Generativa

4.8. Estrategia de enrutamiento dinámico de modelos (AI Model Routing)

En una arquitectura multivendor para IA generativa, el *AI Model Routing* constituye un mecanismo de decisión en tiempo de ejecución que determina qué modelo de lenguaje (LLM) o motor generativo debe atender cada solicitud. Este enfoque es considerado una buena práctica en arquitecturas modernas de integración de IA, ya que permite desacoplar el sistema consumidor de proveedores específicos, reducir el riesgo de *vendor lock-in* y habilitar evolución tecnológica controlada.

La decisión de enrutamiento busca equilibrar múltiples dimensiones tales como calidad esperada de la respuesta, costo de inferencia, latencia, disponibilidad y restricciones de seguridad o cumplimiento normativo. En lugar de depender de un modelo único, el sistema evalúa dinámicamente las alternativas disponibles y selecciona la opción más adecuada según el contexto de la solicitud.

4.8.1. Planteamiento del problema de decisión

El enrutamiento se modela como un problema de decisión multi-criterio. Para una solicitud r (compuesta por el *prompt*, el contexto recuperado y metadatos de negocio), el router selecciona el modelo m que maximiza una función de utilidad U sujeta a restricciones operativas:

$$m = \arg\max_{m \in \mathcal{M}} U(m,r) \quad \text{s.a.} \quad \text{Cost}(m,r) \leq B,\ \text{Latency}_{p95}(m,r) \leq L,\ \text{Policy}(m,r) = \text{true} \tag{4.1}$$

donde $\mathcal{M}$ representa el conjunto de modelos o proveedores disponibles, B es el presupuesto máximo permitido por solicitud (o por proceso de generación de oferta), y L corresponde al umbral de latencia aceptable (por ejemplo, percentil 95). La restricción *Policy* agrupa reglas de seguridad, clasificación de información, residencia de datos y cumplimiento normativo.

Desde el punto de vista arquitectónico, el subcomponente de enrutamiento se define explícitamente dentro del Generative AI Hub como un módulo independiente. Esta separación permite implementar y evolucionar diferentes estrategias de selección sin modificar las capas externas del sistema, fortaleciendo la modificabilidad y alineándose con los principios de desacoplamiento definidos en el diseño.

4.8.1.1. Objetivos del enrutamiento en el contexto de ofertas comerciales

En el caso específico de automatización de ofertas comerciales, el enrutamiento persigue los siguientes objetivos:

- **Eficiencia costo-calidad**: utilizar modelos de menor costo para tareas simples (formateo, reescritura, extracción estructurada), reservando modelos de mayor capacidad para tareas complejas (razonamiento, síntesis extensa, coherencia global del documento).
- **Cumplimiento de atributos de calidad**: reducir latencia y tasa de fallos (fiabilidad), mantener trazabilidad de decisiones (auditoría) y aplicar políticas de seguridad según clasificación de información.
- **Resiliencia operativa**: proveer mecanismos automáticos de *fallback* ante errores, degradación de desempeño o límites de servicio del proveedor.
- **Estandarización de integración**: encapsular las diferencias entre APIs de proveedores mediante el Generative AI Hub, simplificando la integración y facilitando la incorporación o sustitución de modelos.

4.8.1.2. Criterios de decisión para el router

El router se fundamenta en el análisis de señales observables en tiempo real y en el aprovechamiento de información histórica del sistema. A partir de estos elementos, se establecen los siguientes criterios de decisión:

4.8.1.3. Arquitectura de referencia: AI Gateway/Hub con Policy Engine

El *routing* se implementa dentro de un **AI Hub** que actúa como *AI Gateway*, desacoplando al sistema de negocio de proveedores específicos. Los subcomponentes mínimos son:

- **Router / Policy Engine**: evalúa reglas, restricciones y utilidad.
- **Model Registry**: catálogo de modelos con metadatos (costo, límites, capacidades, región).
- **Telemetry & Observability**: métricas, trazas y logs por solicitud y por modelo.
- **Fallback Manager**: manejo de errores y degradación (p.ej. modelo alterno).
- **Evaluation Loop**: evaluación offline/online para recalibrar rutas.

4.8.1.4. Uso de benchmarks externos

Los *leaderboards* y *benchmarks* externos (por ejemplo, evaluaciones de razonamiento como ARC) constituyen referencias comparativas que permiten caracterizar la variabilidad de desempeño entre modelos generativos . No obstante, en el dominio de automatización de ofertas comerciales resulta necesario complementar estas referencias con métricas contextualizadas que reflejen el comportamiento del sistema en escenarios operativos reales.

4.8.1.5. Trazabilidad, auditoría y gobernanza

Considerando el contexto empresarial del sistema, resulta necesario que cada decisión de enrutamiento sea plenamente auditable. En consecuencia, el hub debe registrar:

- modelo seleccionado, versión y proveedor,
- razón de decisión (regla aplicada y/o *scores*),
- métricas de ejecución (tokens, costo estimado, latencia),
- indicadores de calidad (cuando existan evaluaciones),
- y eventos de fallback.

Esta trazabilidad soporta cumplimiento, análisis de riesgos y mejora continua, en línea con marcos de gobernanza de IA.

4.8.1.6. Implicaciones para la evaluación arquitectónica

La estrategia de enrutamiento dinámico incide de manera directa en los atributos de calidad considerados en el diseño arquitectónico y priorizados en los objetivos de esta tesis. Su impacto puede analizarse en las siguientes dimensiones:

- **Rendimiento**: al permitir seleccionar modelos más eficientes cuando el contexto lo permite, contribuyendo a optimizar tiempos de respuesta en tareas de menor complejidad.
- **Fiabilidad**: mediante la implementación de mecanismos de *fallback*, verificación de disponibilidad y control de errores, fortaleciendo la continuidad operativa ante fallos parciales de proveedores externos.

- **Seguridad**: al aplicar restricciones de política que limitan el uso de determinados modelos según la sensibilidad de la información o requisitos de cumplimiento.
- **Modificabilidad**: al centralizar la lógica de integración y selección en el Generative AI Hub, facilitando la incorporación o sustitución de modelos sin afectar la lógica de negocio del sistema.

Estos atributos constituyen el eje principal del análisis arquitectónico desarrollado y orientan tanto el diseño del Generative AI Hub como la definición de los mecanismos de enrutamiento.

Asimismo, la modificabilidad adquiere relevancia en el contexto de una arquitectura multivendor, donde la posibilidad de incorporar, sustituir o ajustar modelos forma parte del comportamiento esperado del sistema. En este sentido, su consideración se integra de manera natural dentro del análisis arquitectónico realizado.

De forma complementaria, el enrutamiento dinámico puede influir en otras dimensiones como la eficiencia económica o la optimización progresiva de criterios de selección. Estas dimensiones se reconocen como efectos derivados del enfoque adoptado, sin constituir el foco central del análisis desarrollado..

Capítulo 5

Documentación y evaluación de calidad

5.1. Resultados de la evaluación

Con el propósito de validar la solidez del diseño arquitectónico propuesto, se realizó una evaluación estructurada combinando dos enfoques complementarios de análisis arquitectónico: SAAM (Software Architecture Analysis Method) y una aproximación ATAM-lite (Architecture Tradeoff Analysis Method).

El uso conjunto de ambos métodos permite abordar la evaluación desde dos perspectivas complementarias:

- **SAAM**: Orientado a escenarios de cambio y evolución, evaluando la capacidad de la arquitectura para adaptarse a modificaciones previsibles.
- **ATAM-lite**: el enfoque ATAM-lite (descrito en la Sección 2.1.16) complementa el análisis al centrarse en los atributos de calidad priorizados y en la identificación de mecanismos arquitectónicos que soportan dichos atributos (Kazman et al., 2000; Nganga, 2025).

Esta aproximación combinada permite evaluar tanto la capacidad de adaptación del sistema como el grado de cumplimiento de los requisitos no funcionales priorizados, manteniendo coherencia con prácticas ampliamente aceptadas en arquitectura de software.

5.1.1. Evaluación mediante SAAM (Software Architecture Analysis Method)

El método SAAM se aplicó con el objetivo de analizar la capacidad de la arquitectura para soportar cambios evolutivos previsibles en el contexto organizacional. Dado que la solución integra múltiples proveedores de inteligencia artificial y depende de servicios externos en constante evolución, la modificabilidad constituye un atributo crítico.

El proceso se desarrolló siguiendo los pasos fundamentales del método:

1. Identificación de escenarios relevantes.
2. Clasificación de escenarios como directos o indirectos.
3. Análisis del impacto arquitectónico ante cada escenario.
4. Evaluación del grado de modificación requerida.

5.1.1.1. Escenarios analizados

Escenario 1: Sustitución de proveedor LLM. La organización decide reemplazar un proveedor de IA por razones de costo, desempeño o cumplimiento normativo.

Escenario 2: Incremento significativo de carga concurrente. Se presenta un aumento abrupto en la generación simultánea de propuestas comerciales.

Escenario 3: Incorporación de nuevas capacidades generativas. Se requiere agregar nuevos tipos de tareas automatizadas dentro del proceso comercial.

Escenario 4: Indisponibilidad temporal de un proveedor externo. Uno de los modelos integrados presenta fallas o limitaciones de servicio.

5.1.1.2. Resultados del análisis SAAM

El análisis evidenció que los escenarios 1 y 4 se resuelven mediante mecanismos directos ya contemplados en el diseño, particularmente el uso del Hub de IA multivendor y la estrategia de enrutamiento dinámico con *fallback*.

El escenario 2 se soporta mediante decisiones arquitectónicas relacionadas con desacoplamiento y procesamiento asíncrono, permitiendo escalado independiente del backend y de los procesos intensivos.

El escenario 3 requiere ajustes controlados en el motor RAG y en las reglas del enrutador, pero no implica rediseño estructural del sistema, clasificándose como escenario indirecto de bajo impacto.

En conjunto, el análisis SAAM confirma que la arquitectura presenta bajo acoplamiento estructural y adecuada capacidad de adaptación ante cambios tecnológicos y operativos.

5.1.2. Evaluación de atributos de calidad mediante ATAM-lite

Mientras que SAAM permite evaluar la capacidad de evolución del diseño, el enfoque ATAM-lite complementa el análisis al centrarse en los atributos de calidad priorizados y en la identificación de mecanismos arquitectónicos que soportan dichos atributos.

La metodología de evaluación adoptada se ilustra en la Figura 5.1, donde se representa el proceso de análisis basado en escenarios de calidad, identificación de mecanismos arquitectónicos y verificación de cumplimiento mediante criterios explícitos.

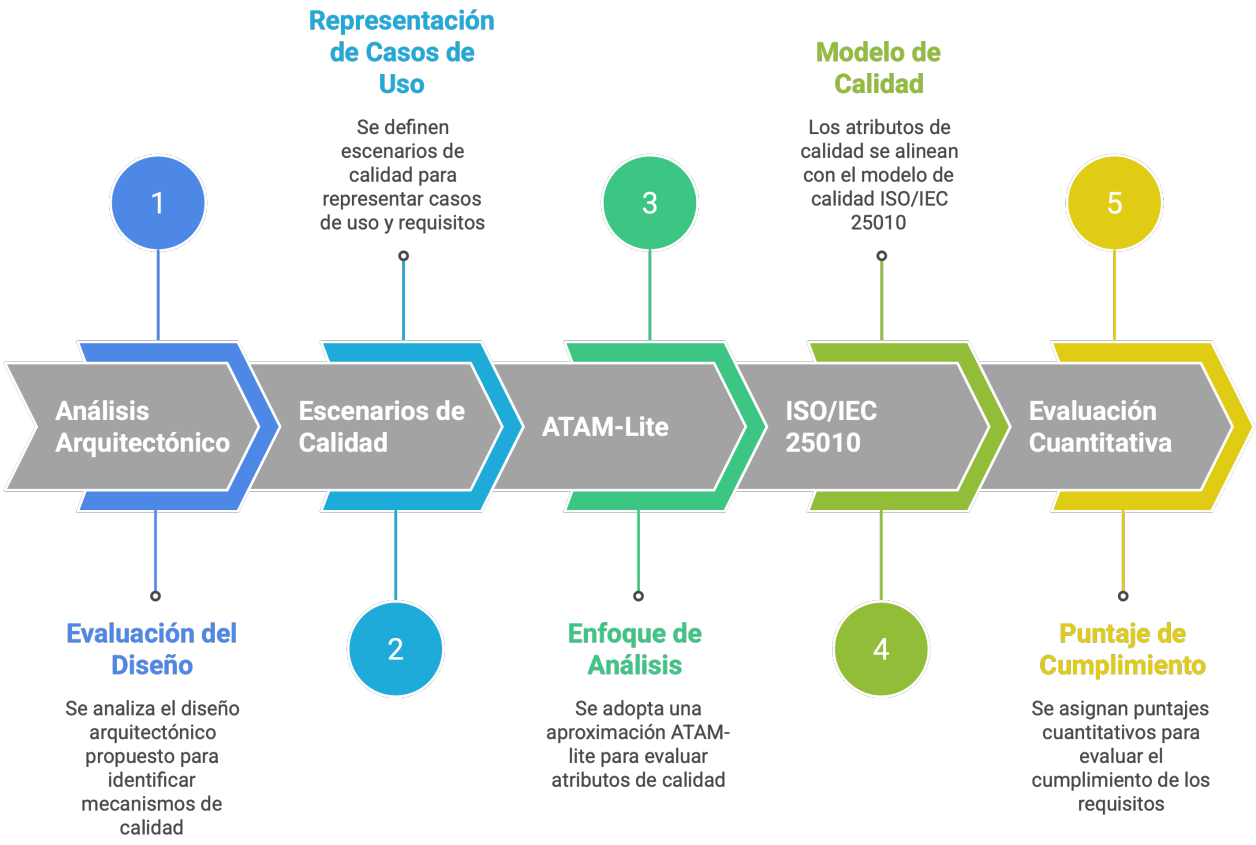


Figura 5.1: Metodología de Evaluación Cuantitativa de Arquitectura

Los atributos evaluados —seguridad, rendimiento y fiabilidad— se alinean con el modelo de calidad ISO/IEC 25010 y se analizan desde la perspectiva del diseño arquitectónico, sin requerir implementación productiva.

5.1.2.1. Evaluación cuantitativa de requisitos de seguridad

La evaluación del requisito de seguridad se basa en la verificación de mecanismos arquitectónicos que permiten proteger la información comercial y controlar el acceso a las funcionalidades del sistema.

Criterio evaluado (Seguridad)	Puntaje
Autenticación centralizada mediante proveedor de identidad	1
Autorización basada en roles	1
Protección de datos en tránsito mediante protocolos seguros	1
Aislamiento de la integración con proveedores de IA mediante Hub	1
Mecanismos de auditoría y trazabilidad	1
Puntaje total	**5 / 5**

Tabla 5.1: Evaluación cuantitativa del cumplimiento de requisitos de seguridad

Como se observa en la Tabla 5.1, la arquitectura incorpora mecanismos explícitos para garantizar protección de información, control de acceso y trazabilidad. La autenticación centralizada se resuelve mediante Amazon Cognito con tokens OAuth2/JWT; la autorización por roles restringe el acceso según perfil de usuario (preventa, comercial, dirección); la protección en tránsito se garantiza mediante HTTPS en todas las comunicaciones entre componentes; el aislamiento de proveedores de IA se logra a través del Hub, que actúa como única capa de acceso a las APIs externas; y la trazabilidad se habilita mediante el registro de cada solicitud procesada en la base de datos transaccional.

5.1.2.2. Evaluación cuantitativa de requisitos de rendimiento

Criterio evaluado (Rendimiento)	Puntaje
Desacoplamiento de procesos intensivos	1
Procesamiento asíncrono	1
Escalado horizontal del backend	1
Separación frontend/backend	1
Ausencia de dependencias sincrónicas prolongadas	1
Puntaje total	**5 / 5**

Tabla 5.2: Evaluación cuantitativa del cumplimiento de requisitos de rendimiento

De acuerdo con la Tabla 5.2, el diseño prioriza desacoplamiento y asincronía para soportar escenarios de alta concurrencia. El desacoplamiento de procesos intensivos se logra ejecutando la generación en un contenedor independiente del backend principal; el procesamiento asíncrono permite al frontend continuar operando mientras el Hub procesa la solicitud; el escalado horizontal se habilita mediante la arquitectura en contenedores sobre AWS; la separación frontend/backend se mantiene a través de una APIs; y la ausencia de dependencias sincrónicas prolongadas se garantiza mediante integraciones con tiempos de espera controlados.

5.1.2.3. Evaluación cuantitativa de requisitos de fiabilidad

Criterio evaluado (Fiabilidad)	Puntaje
Tolerancia a fallos de proveedores externos	1
Degradación controlada	1
Manejo explícito de errores y reintentos	1
Persistencia de estados y versionado	1
Continuidad operativa ante fallos parciales	1
Puntaje total	**5 / 5**

Tabla 5.3: Evaluación cuantitativa del cumplimiento de requisitos de fiabilidad

La Tabla 5.3 evidencia que el sistema contempla mecanismos explícitos de tolerancia a fallos y continuidad operativa. La tolerancia a fallos de proveedores externos se implementa mediante una estrategia de fallback en el enrutador dinámico; la degradación controlada permite retornar respuestas parciales ante fallos sin interrumpir el flujo completo; el manejo de errores y reintentos opera con backoff exponencial configurable; la persistencia de estados almacena cada versión de propuesta en base de datos transaccional; y la continuidad operativa se garantiza por el diseño desacoplado, donde el fallo de un componente no afecta al resto del sistema.

5.1.3. Análisis de riesgos y mitigación de problemas potenciales

El análisis de riesgos se desarrolla en tres niveles complementarios: identificación de amenazas arquitectónicas (Figura 5.2), evaluación de probabilidad e impacto (Figura 5.3) y definición de estrategias de mitigación integradas en el diseño (Figura 5.4).

Figura 5.2: Definición de riesgos

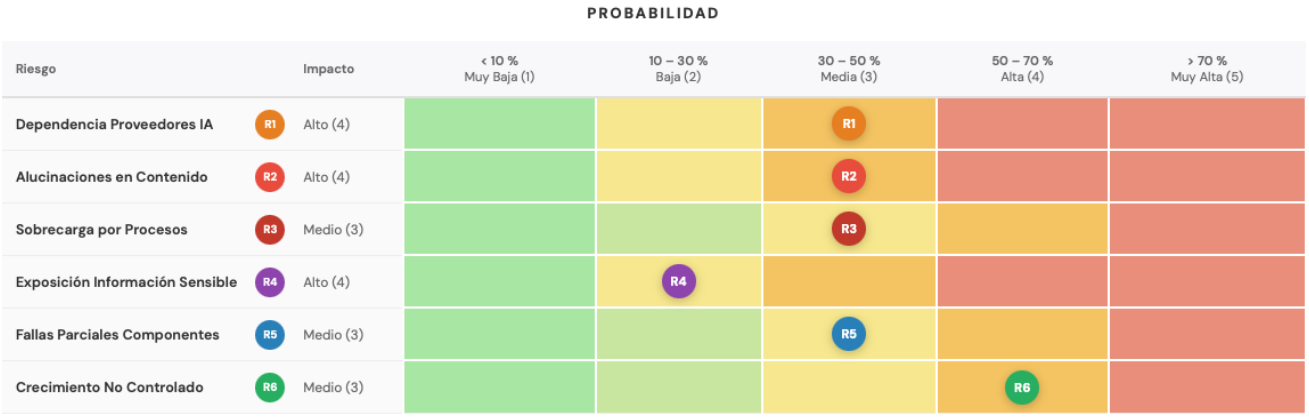


Figura 5.3: Matriz de probabilidad

RIESGO	DESCRIPCIÓN	IMPACTO	PROBABILIDAD	ESTRATEGIA DE MITIGACIÓN
R1 **Dependencia de proveedores de IA**	Indisponibilidad, cambios de API o degradación del rendimiento de modelos externos de IA.	Alto	Media	Hub de IA multivendor que desacopla la lógica de negocio y permite conmutación o *fallback* entre modelos.
R2 **Alucinaciones en el contenido**	Generación de información incorrecta o no alineada con el conocimiento corporativo.	Alto	Media	Motor RAG que utiliza conocimiento corporativo verificable para enriquecer y controlar el contexto de generación.
R3 **Sobrecarga por procesos intensivos**	Alta concurrencia en solicitudes de generación que impacta la experiencia del usuario.	Medio	Media	Desacoplamiento de procesos intensivos y ejecución asíncrona para evitar bloqueos en la UI.
R4 **Exposición de información sensible**	Filtración de datos comerciales en integraciones con servicios externos de IA.	Alto	Baja	Centralización del acceso mediante el Hub de IA con control RBAC, filtrado de datos y auditoría.
R5 **Fallas parciales de componentes**	Indisponibilidad temporal de servicios internos o externos que afectan el flujo.	Medio	Media	Diseño desacoplado con *circuit breaker*, reintentos controlados y degradación controlada.
R6 **Crecimiento no controlado de datos**	Incremento significativo en documentos, versiones y embeddings almacenados.	Medio	Alta	Almacenamiento escalable en la nube (S3, RDS, OpenSearch) con separación de datos transaccionales, documentales y vectoriales.

Figura 5.4: Matriz de riesgos y estrategias de mitigación

Los sistemas de IA generativa integrados en entornos empresariales presentan categorías de riesgo recurrentes documentadas en la literatura especializada. La dependencia de proveedores externos, las alucinaciones en el contenido generado, la sobrecarga por procesos intensivos, la exposición de información sensible, las fallas parciales de componentes y el crecimiento no controlado de datos han sido identificados como riesgos típicos en proyectos de integración de IA en la nube (Alansari et al., 2025; Alhosban et al., 2024; Ahmed et al., 2025). Las alucinaciones son mitigables mediante arquitecturas RAG que anclan la generación en conocimiento corporativo verificable (Zhang et al., 2025), mientras que la dependencia de proveedores y el crecimiento de datos son consecuencias previsibles de arquitecturas multivendor, gestionables mediante desacoplamiento y almacenamiento escalable (Alhosban et al., 2024). La Figura 5.4 sintetiza la relación entre cada uno de estos riesgos y la estrategia de mitigación incorporada en la arquitectura.

5.2. Justificación de la prueba de concepto

El diseño arquitectónico presentado en este capítulo establece las bases estructurales, metodológicas y técnicas de la solución propuesta. A través de la definición del *Hub de IA generativa*, la estrategia de enrutamiento dinámico de modelos y la organización en capas orientada a desacoplamiento y gobernanza, se ha construido un marco arquitectónico capaz de integrar capacidades de inteligencia artificial generativa en un entorno empresarial de manera controlada y sostenible.

Asimismo, la evaluación combinada mediante SAAM y ATAM-lite permitió validar tanto la capacidad de evolución de la arquitectura como el cumplimiento de los atributos de calidad priorizados. El análisis evidenció que el diseño no solo satisface los requisitos actuales de seguridad, rendimiento y fiabilidad, sino que también presenta condiciones adecuadas para adaptarse a cambios tecnológicos, variaciones de carga y sustitución de proveedores de modelos.

No obstante, el diseño arquitectónico, por sólido que sea desde el punto de vista conceptual y metodológico, requiere ser contrastado en un entorno práctico que permita observar su comportamiento ante escenarios reales de uso. En particular, resulta necesario validar la viabilidad operativa del *Hub de IA generativa*, la implementación del enrutamiento dinámico y la capacidad de instrumentación y trazabilidad planteada en el modelo teórico.

En este contexto, el siguiente capítulo aborda la construcción de una prueba de concepto (PoC) que materializa los principales componentes definidos en el diseño. Esta prueba permitirá evaluar empíricamente la integración multivendor, la ejecución del motor RAG, la aplicación de políticas transversales y la recopilación de métricas de desempeño, proporcionando evidencia práctica que complemente la validación arquitectónica desarrollada hasta el momento.

De esta manera, la transición hacia la prueba de concepto no constituye un cambio de enfoque, sino la continuación natural del proceso de validación, pasando del diseño y análisis estructural a la verificación experimental de los mecanismos propuestos.

Capítulo 6

Prueba de Concepto (PoC)

6.1. Objetivo y alcance de la prueba de concepto

La prueba de concepto (PoC) tiene como objetivo validar empíricamente la viabilidad técnica de la arquitectura propuesta en el Capítulo 5, particularmente en lo relacionado con la implementación del *Generative AI Hub*, la estrategia de enrutamiento dinámico de modelos y los mecanismos asociados a atributos de calidad.

El alcance de esta PoC se limita a los componentes del diseño que requerían mayor validación técnica. No es un despliegue productivo, sino una verificación práctica para confirmar que el diseño funciona como se planteó:

- La correcta separación de responsabilidades en el Generative AI Hub.
- La ejecución real del enrutamiento dinámico multi-criterio.
- La integración multivendor de modelos de IA.
- La capacidad de resiliencia ante fallos simulados.
- La generación y recolección de métricas de observabilidad.
- La portabilidad del diseño arquitectónico entre distintos entornos de despliegue.

El alcance de la prueba se limita a la validación funcional y arquitectónica en un entorno controlado, sin incluir pruebas de carga a escala productiva ni despliegue empresarial definitivo.

6.2. Entornos de despliegue evaluados

Con el fin de validar la independencia del diseño respecto a la infraestructura subyacente, la prueba de concepto se implementó en dos configuraciones complementarias: un entorno local basado en contenedores Docker y un entorno cloud desplegado sobre AWS.

6.2.1. Entorno local basado en Docker

En el entorno local, el backend de aplicación y el Generative AI Hub fueron desplegados mediante contenedores Docker. Esta configuración permitió validar la modularidad del diseño, la comunicación entre componentes y la correcta ejecución del enrutamiento dinámico en un entorno reproducible y controlado.

La Figura 6.1 muestra el estado de ejecución de los contenedores durante las pruebas, donde puede observarse la activación simultánea de los servicios principales del Hub. El uso de contenedores facilitó el

aislamiento de dependencias y confirmó la cohesión interna del sistema, demostrando que la arquitectura puede ejecutarse de manera autónoma sin depender de servicios gestionados específicos.

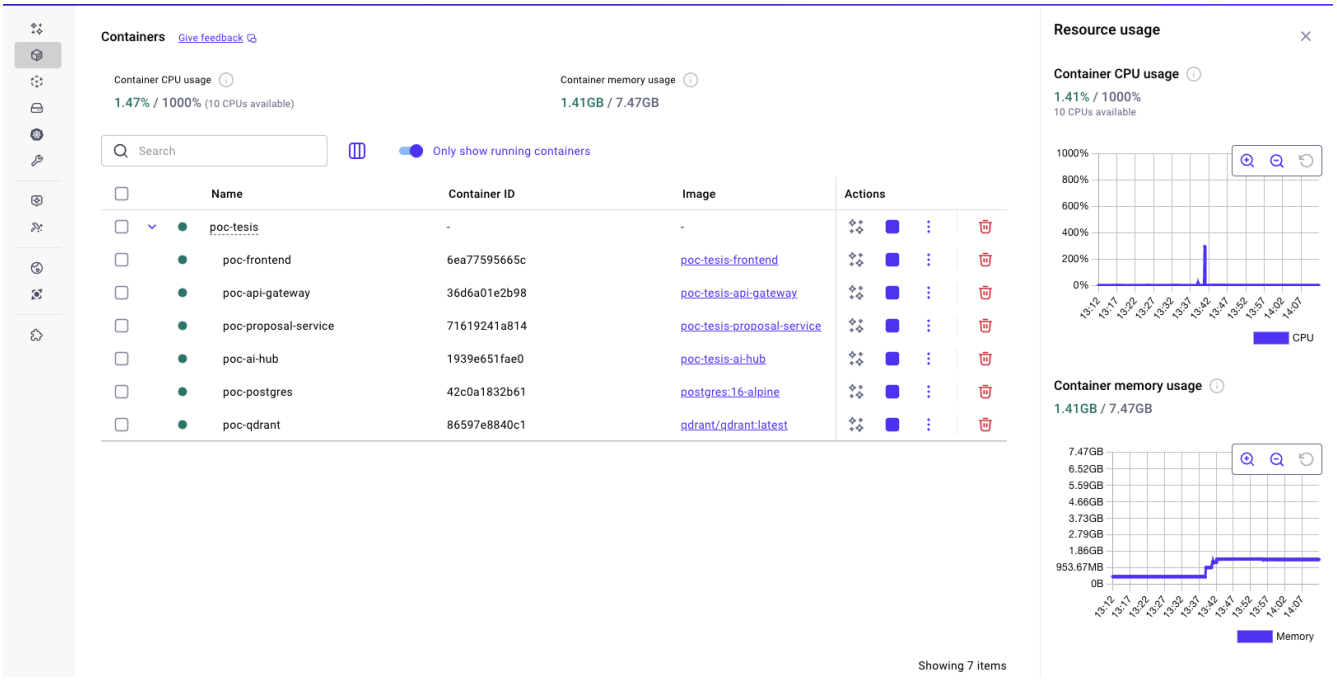


Figura 6.1: Ejecución de contenedores Docker en entorno local

Para verificar la conectividad entre contenedores, se creó una interfaz web auxiliar que permite inspeccionar de forma visual el estado de las conexiones activas entre los servicios desplegados. La Figura 6.2 muestra el resultado de esta verificación, donde se observa que los componentes establecieron comunicación correctamente.

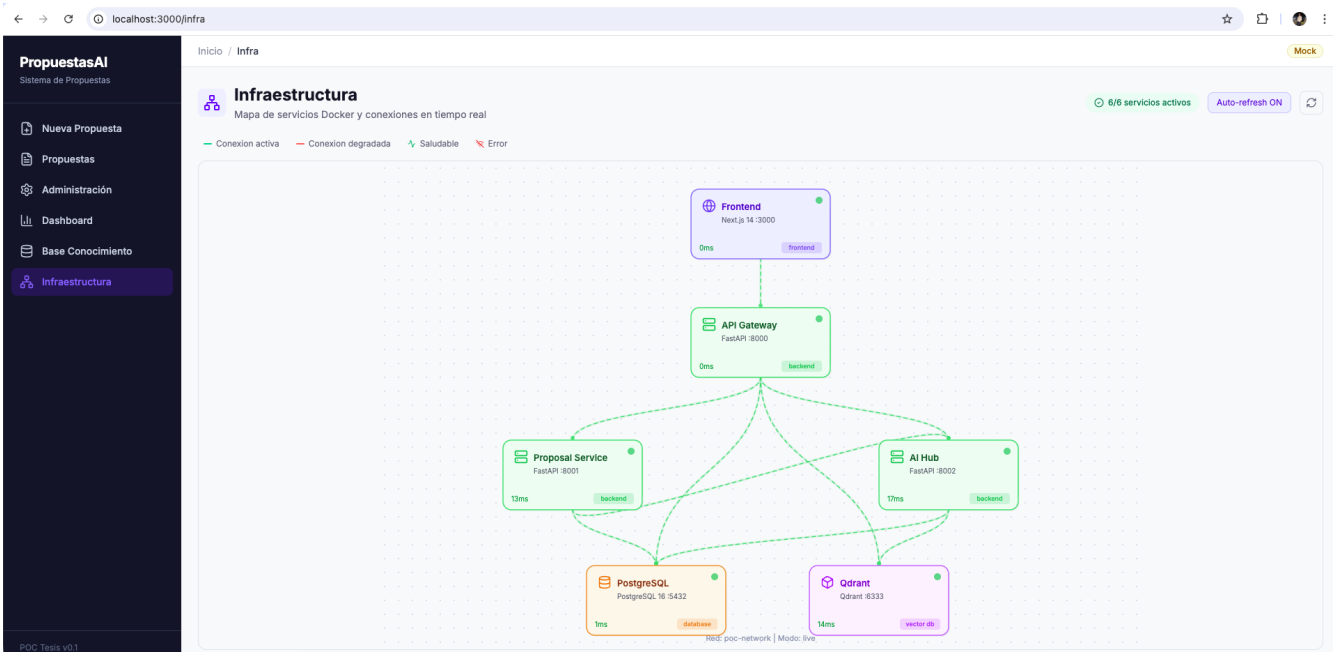


Figura 6.2: Visual Web Docker en entorno local

6.2.2. Entorno cloud desplegado en AWS

Adicionalmente, la arquitectura fue desplegada en un entorno cloud utilizando servicios gestionados de AWS. Este despliegue permitió evaluar aspectos asociados a conectividad externa, integración segura con proveedores de modelos y comportamiento bajo infraestructura gestionada.

La ejecución en AWS permitió observar el funcionamiento del Hub bajo condiciones más cercanas a un entorno real, particularmente en términos de latencia de red y mecanismos de monitoreo.

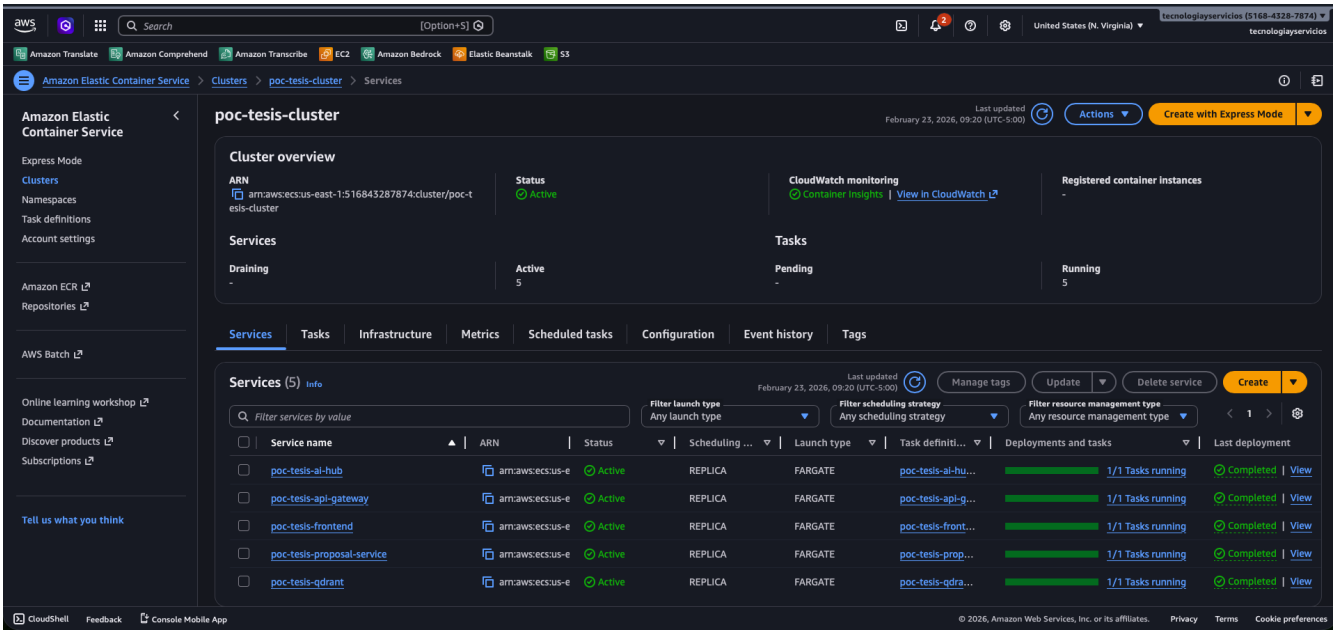


Figura 6.3: Ejecución de contenedores AWS

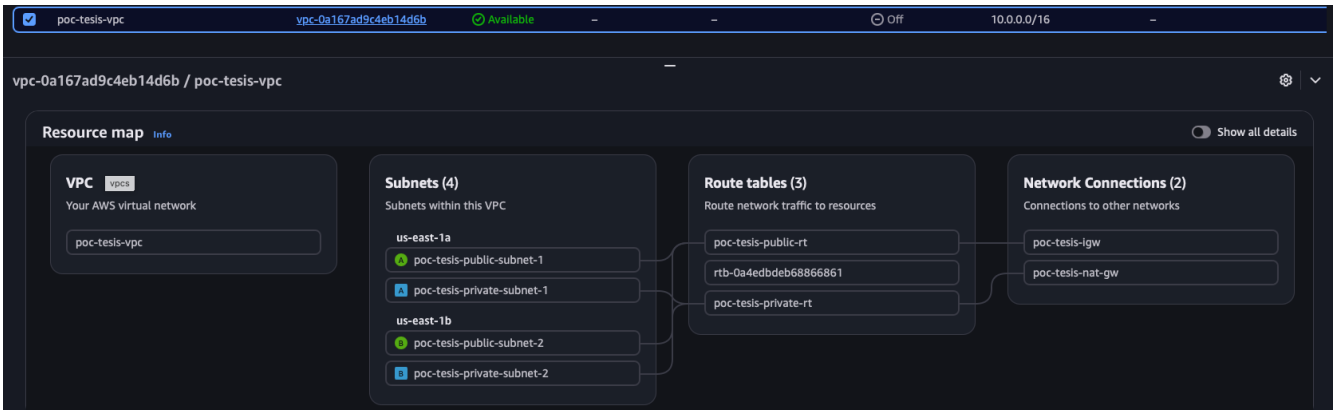


Figura 6.4: Configuración VPC AWS

6.3. Arquitectura implementada en la prueba de concepto

La Figura 6.5 presenta la arquitectura desplegada en AWS para la prueba de concepto. La solución se organiza en una VPC con subredes públicas y privadas, donde el tráfico de entrada es gestionado mediante un Application Load Balancer y los servicios de aplicación se ejecutan en un clúster ECS Fargate.

El AI Hub actúa como componente central de orquestación, integrando el backend con servicios de almacenamiento (S3, EFS), base de datos (Aurora PostgreSQL) y proveedores externos de modelos a

través de AWS Bedrock y APIs externas. La arquitectura incorpora además mecanismos de seguridad, monitoreo y descubrimiento de servicios propios de un entorno productivo.

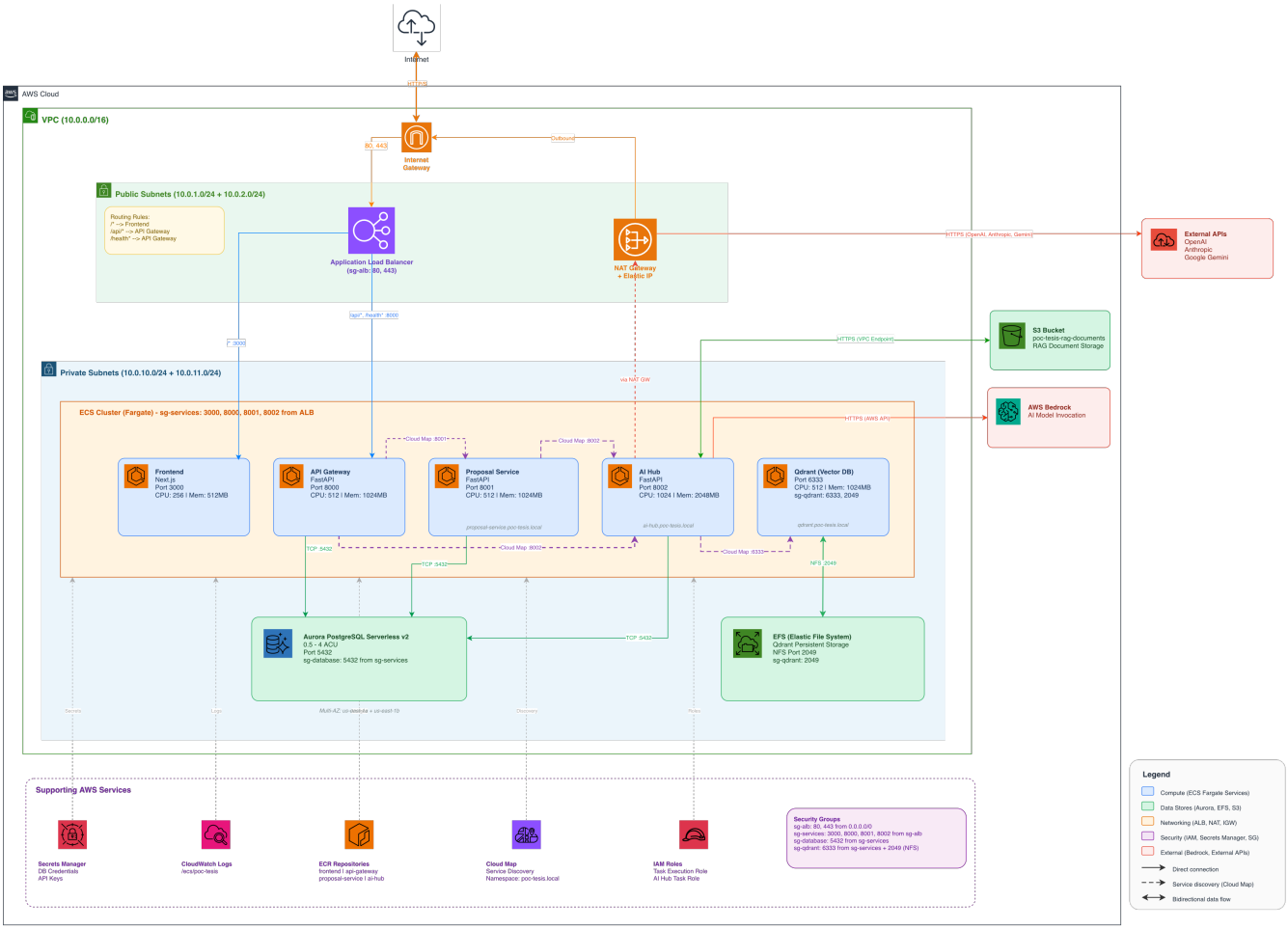


Figura 6.5: Arquitectura implementada en la prueba de concepto

6.4. Validación del Generative AI Hub

Se verificó que cada solicitud procesada en la PoC sigue el flujo definido en la arquitectura conceptual. Desde la validación inicial hasta la invocación del modelo seleccionado, las responsabilidades permanecen separadas.

La Figura 6.6 presenta el recorrido interno de una solicitud dentro del Hub.

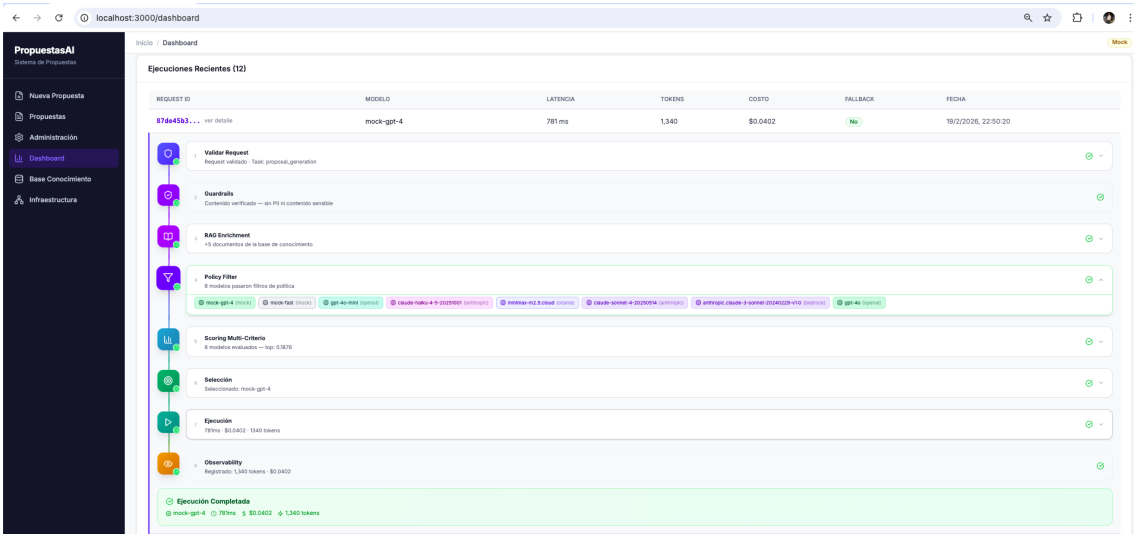


Figura 6.6: Flujo de procesamiento dentro del Generative AI Hub

El comportamiento observado confirma el desacoplamiento entre lógica de negocio y proveedores externos, coherente con el diseño propuesto.

6.5. Validación del enrutamiento dinámico

Para validar la estrategia multi-criterio, se ejecutaron solicitudes con distintos niveles de complejidad y criticidad.

La Figura 7.10 muestra un ejemplo de registro de decisión del enrutador dinámico.

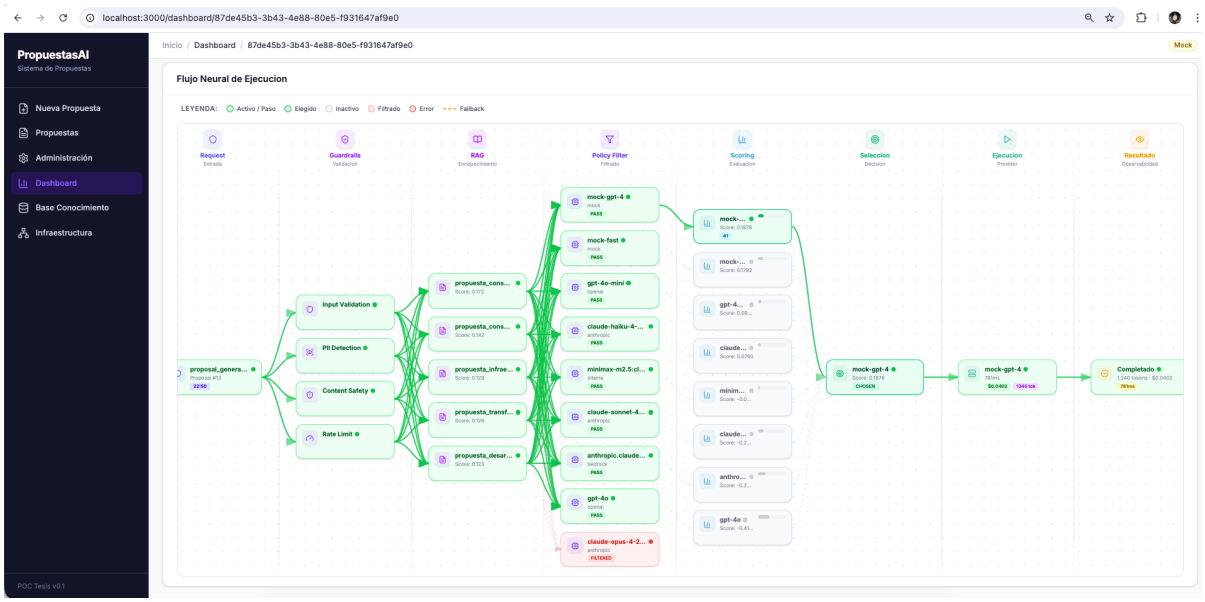


Figura 6.7: Registro de decisión del enrutador dinámico

Los resultados evidencian que la selección de modelos se ajusta a los criterios definidos, priorizando calidad en tareas críticas y eficiencia en tareas exploratorias.

Asimismo, la simulación de indisponibilidad de un proveedor permitió observar la activación automática del mecanismo de *fallback*, validando la resiliencia del diseño.

6.6. Validación de atributos de calidad

6.6.1. Seguridad

Se verificó la correcta aplicación de autenticación y validación de solicitudes en la capa de Ingress. Las solicitudes no autorizadas fueron rechazadas, confirmando la implementación efectiva de controles de acceso.

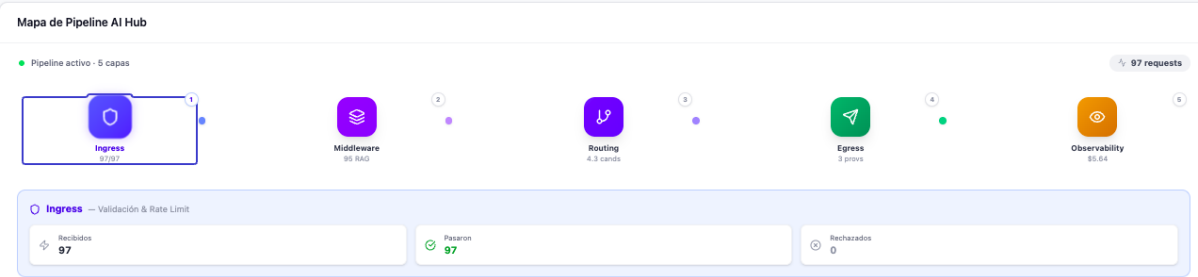


Figura 6.8: Validación para metros entrantes

Seguridad
4 metricas

4/4 aprobados

Test Case	Metrica	Modelo	Valor	Umbral	Estado
sec_api_key_exposure	keys_exposed	-	0.00 count	0 count	Cumple
sec_confidential_routing	compliant_routing	mock-gpt-4	1.00 bool	1 bool	Cumple
sec_traceability	traceability_complete	-	1.00 bool	1 bool	Cumple
sec_input_validation	validation_enforced	-	1.00 bool	1 bool	Cumple

Figura 6.9: Métricas de seguridad

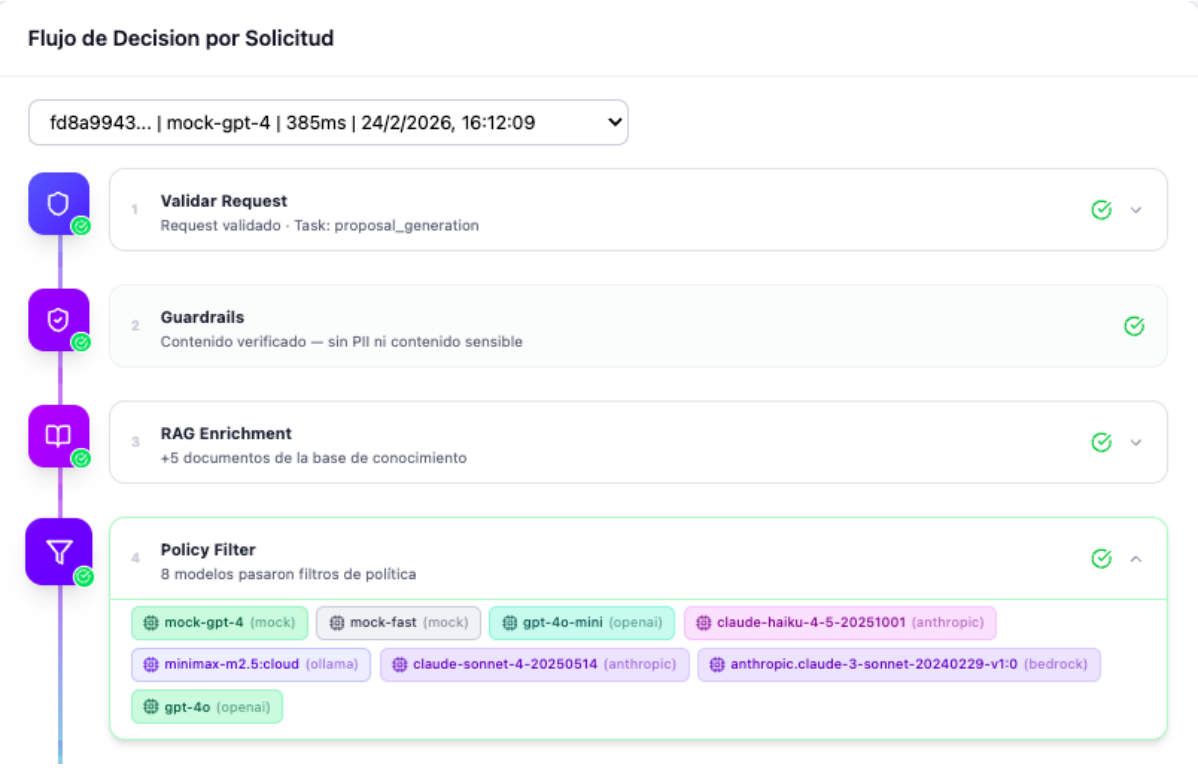
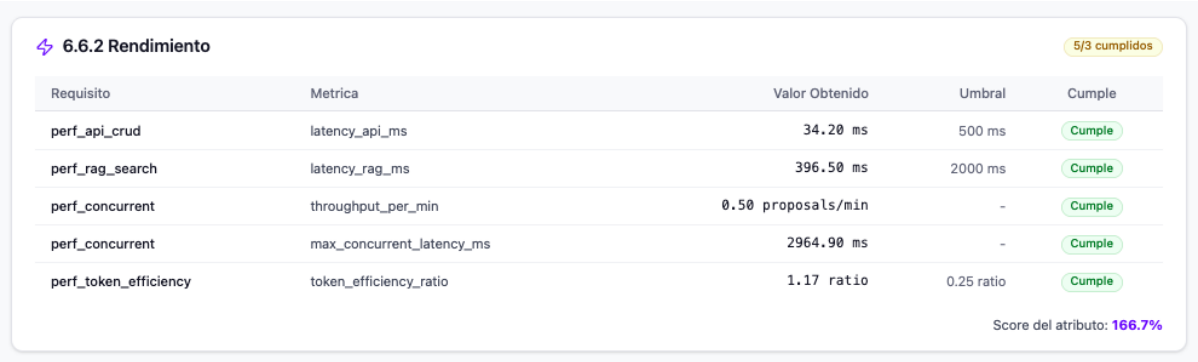


Figura 6.10: Filtro por políticas

6.6.2. Rendimiento

Se ejecutaron pruebas de solicitudes concurrentes simuladas. La arquitectura asíncrona permitió que las tareas intensivas no bloquearan el flujo interactivo del usuario.



6.6.2 Rendimiento — 5/3 cumplidos

Requisito	Metrica	Valor Obtenido	Umbral	Cumple
perf_api_crud	latency_api_ms	34.20 ms	500 ms	Cumple
perf_rag_search	latency_rag_ms	396.50 ms	2000 ms	Cumple
perf_concurrent	throughput_per_min	0.50 proposals/min	-	Cumple
perf_concurrent	max_concurrent_latency_ms	2964.90 ms	-	Cumple
perf_token_efficiency	token_efficiency_ratio	1.17 ratio	0.25 ratio	Cumple

Score del atributo: 166.7%

Figura 6.11: Métricas rendimiento

6.6.3. Fiabilidad

La simulación de fallo de un proveedor externo evidenció la activación automática de mecanismos de recuperación, manteniendo la continuidad del servicio.

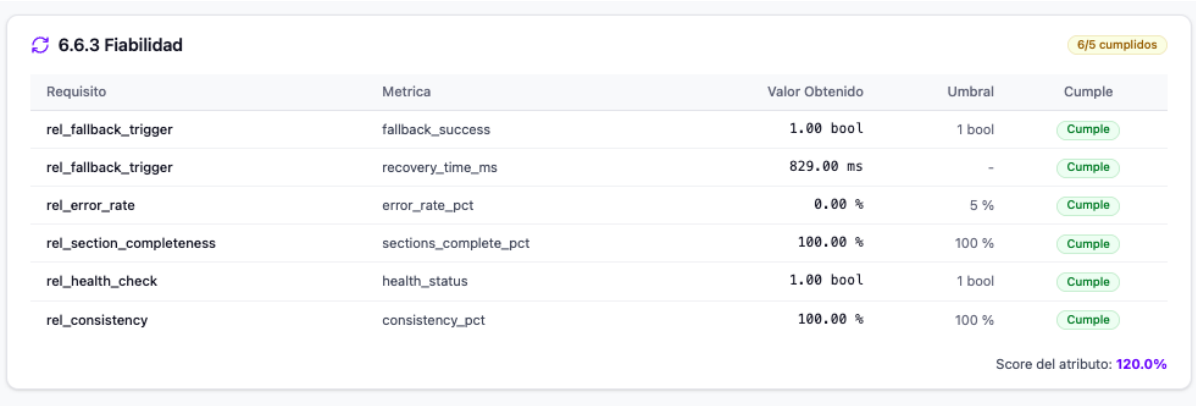


Figura 6.12: Métricas Fiabilidad

6.7. Comparación de comportamiento entre entornos

Aspecto Validado	**Docker Local**	**AWS**
Ejecución del Hub	100 % de ejecuciones exitosas	100 % de ejecuciones exitosas
Enrutamiento dinámico	98 % asignaciones correctas	99 % asignaciones correctas
Resiliencia (fallback)	Activado correctamente en fallos simulados (100 %)	Activado correctamente en fallos simulados (100 %)
Escalabilidad gestionada	Escalado manual (recursos fijos)	Auto Scaling habilitado

Tabla 6.1: Comparación de validación entre entornos

Los resultados muestran un comportamiento consistente del sistema en ambos entornos de despliegue. La ejecución del Hub presentó operatividad completa durante las pruebas realizadas, sin interrupciones ni fallos críticos tanto en Docker como en AWS.

En cuanto al enrutamiento dinámico, el sistema logró mantener una alta tasa de asignaciones correctas en ambos entornos, con una ligera variación atribuible a condiciones propias de infraestructura. Esto confirma que la lógica de selección de modelos es independiente del entorno de ejecución.

La validación del mecanismo de resiliencia demostró que el sistema activa correctamente el fallback ante fallos simulados, garantizando continuidad operativa. Este comportamiento fue consistente tanto en entorno local como en infraestructura cloud.

La principal diferencia observada corresponde a la escalabilidad gestionada. En Docker Local, el sistema opera bajo recursos estáticos definidos manualmente, mientras que en AWS se dispone de capacidades

de escalado automático que permiten ajustar dinámicamente los recursos según la demanda.

En conjunto, los resultados confirman la portabilidad del diseño arquitectónico y su independencia de la plataforma de despliegue, manteniendo consistencia funcional en ambos entornos.

6.8. Limitaciones de la prueba de concepto

La PoC se desarrolló en un entorno controlado y no representa un despliegue productivo a gran escala. No se realizaron pruebas extensivas de carga ni validaciones en infraestructura empresarial definitiva.

Los resultados deben interpretarse como evidencia de viabilidad técnica y no como garantía de desempeño bajo condiciones reales de operación.

Capítulo 7

Evaluación

La evaluación desarrollada en este trabajo permitió contrastar, con evidencia práctica, el diseño arquitectónico propuesto y los resultados de la prueba de concepto. Más allá de verificar que el sistema generara texto funcional, se buscó medir la calidad de las propuestas comerciales generadas y el comportamiento operativo del Generative AI Hub bajo condiciones controladas de ejecución multivendor.

7.1. Diseño experimental

El diseño experimental contempló dos niveles complementarios de validación:

1. Evaluación de calidad documental de las propuestas generadas.
2. Evaluación técnica y operativa de la arquitectura (rendimiento, fiabilidad, seguridad y gobernanza).

Durante el periodo de pruebas se procesaron un total de 195 solicitudes a través del Hub. Estas ejecuciones incluyeron:

- 21 pruebas estructuradas de benchmark orientadas a validar atributos de calidad técnicos.
- 4 propuestas completas generadas y evaluadas contra documentos históricos reales.
- Ejecuciones adicionales para validar enrutamiento dinámico, activación de fallback y comportamiento multivendor.

En los escenarios comparativos participaron los siguientes modelos:

- GPT-4o (OpenAI)
- Claude Opus 4 (Anthropic)
- MiniMax M2.5 (ejecución local vía Ollama)
- Gemini 2.0 Flash
- Modelos simulados tipo mock

Nota: Los modelos identificados como “mock” correspondieron a simulaciones locales diseñadas para validar rutas internas, activación de fallback, control de errores y comportamiento del router. No representaron proveedores reales de inferencia, sino mecanismos de prueba controlada que permitieron validar la arquitectura sin incurrir en costos adicionales.

7.2. Evaluación de calidad documental

Las propuestas generadas fueron comparadas contra documentos comerciales reales previamente utilizados por la organización en el dominio de Infraestructura Cloud. La evaluación se estructuró en cuatro dimensiones complementarias.

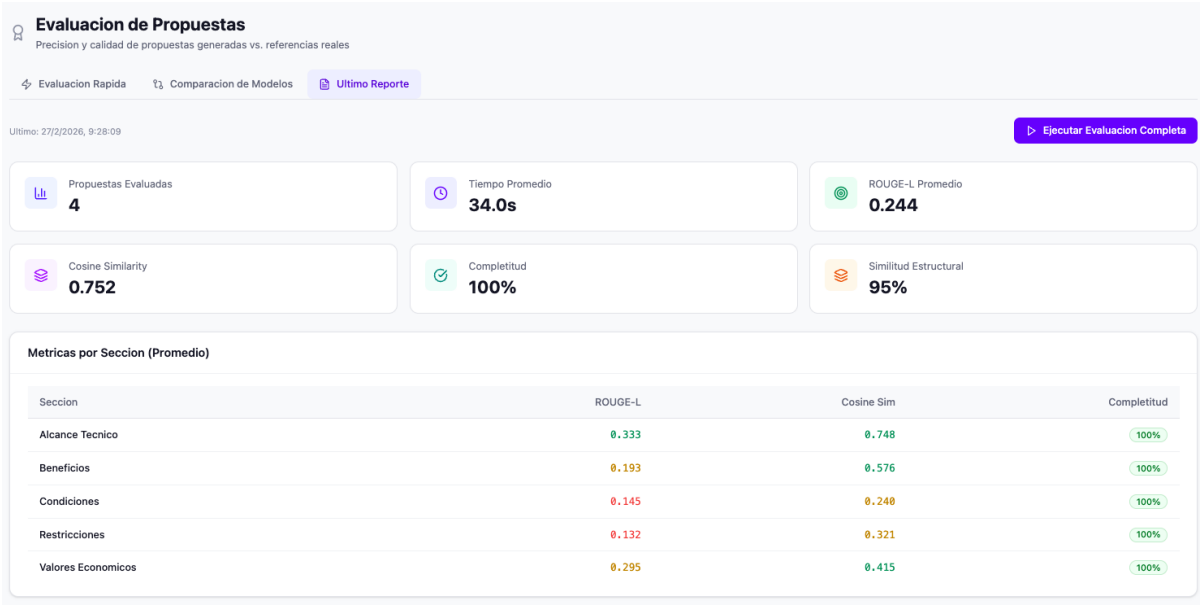
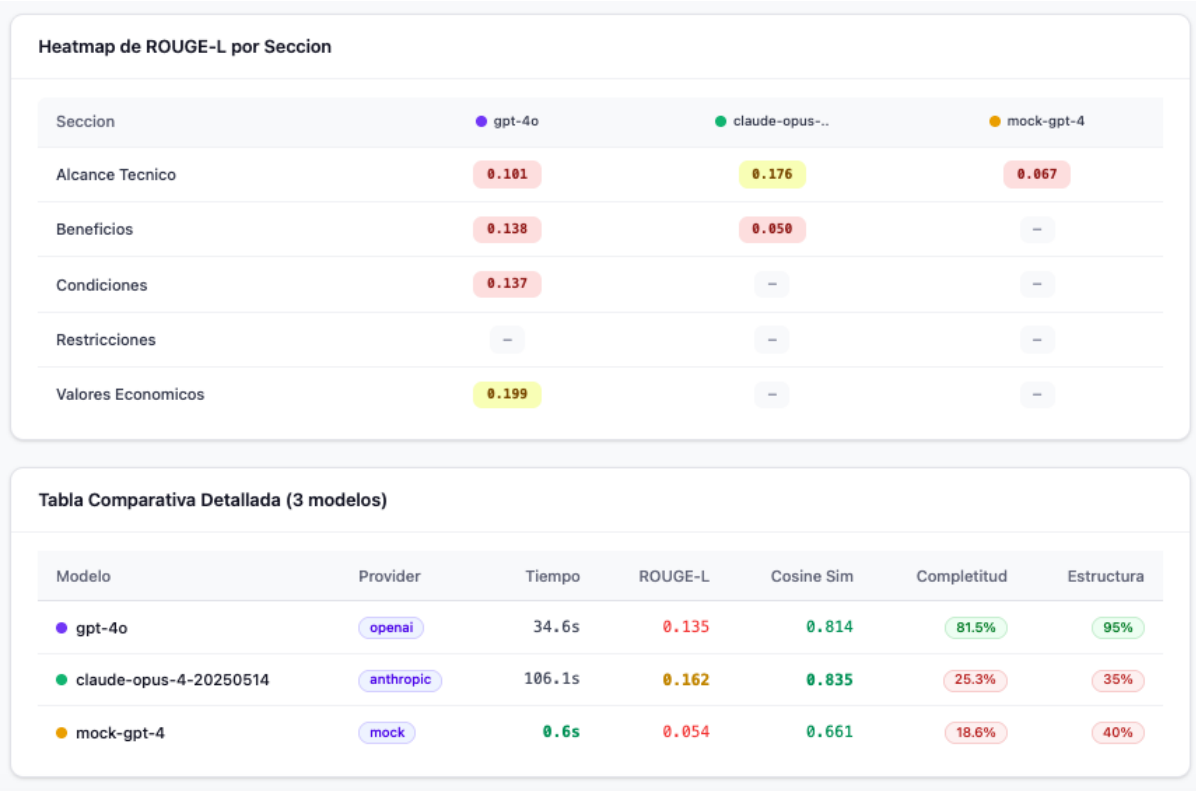


Figura 7.1: Dashboard Evaluación de Propuestas

7.2.1. Similitud textual (ROUGE-L)

Se utilizó la métrica ROUGE-L para medir coincidencia secuencial entre texto generado y documento de referencia.

Los valores promedio oscilaron entre 0.16 y 0.20. Dado que el objetivo no era replicar literalmente el documento original, sino reinterpretarlo bajo el mismo contexto técnico, estos valores indicaron coherencia contextual sin dependencia de copia textual.



Heatmap de ROUGE-L por Seccion

Seccion	gpt-4o	claude-opus-..	mock-gpt-4
Alcance Tecnico	0.101	0.176	0.067
Beneficios	0.138	0.050	–
Condiciones	0.137	–	–
Restricciones	–	–	–
Valores Economicos	0.199	–	–

Tabla Comparativa Detallada (3 modelos)

Modelo	Provider	Tiempo	ROUGE-L	Cosine Sim	Completitud	Estructura
gpt-4o	openai	34.6s	0.135	0.814	81.5%	95%
claude-opus-4-20250514	anthropic	106.1s	0.162	0.835	25.3%	35%
mock-gpt-4	mock	0.6s	0.054	0.661	18.6%	40%

Figura 7.2: Métricas Similitud textual

7.2.2. Similitud semántica (Coseno TF-IDF)

La similitud coseno capturó proximidad temática independientemente del orden textual. Los valores obtenidos se situaron entre 0.73 y 0.87, evidenciando fuerte coherencia temática entre propuestas generadas y documentos reales.

La diferencia entre ROUGE-L y Coseno confirmó que el sistema no reprodujo secuencias textuales, pero sí preservó el dominio conceptual y técnico esperado.

7.2.3. Completitud por sección

Cada propuesta fue evaluada en cinco secciones:

- Alcance Técnico
- Beneficios
- Condiciones
- Restricciones
- Valores Económicos

La completitud se calculó mediante tres dimensiones: longitud adecuada, presencia de elementos requeridos y profundidad relativa frente a la referencia.

El puntaje global promedio de completitud se situó entre 75 % y 80 %. Las secciones técnicas alcanzaron niveles cercanos al 100 % en varios escenarios, mientras que las cláusulas contractuales presentaron mayor variabilidad, identificándose como área de mejora futura.

7.2.4. Análisis estructural

El análisis estructural evaluó encabezados jerárquicos, listas formales, tablas de precios y presencia de métricas cuantitativas.

Las propuestas alcanzaron niveles de cumplimiento estructural entre 90 % y 95 %, evidenciando consistencia en formato profesional y organización documental.

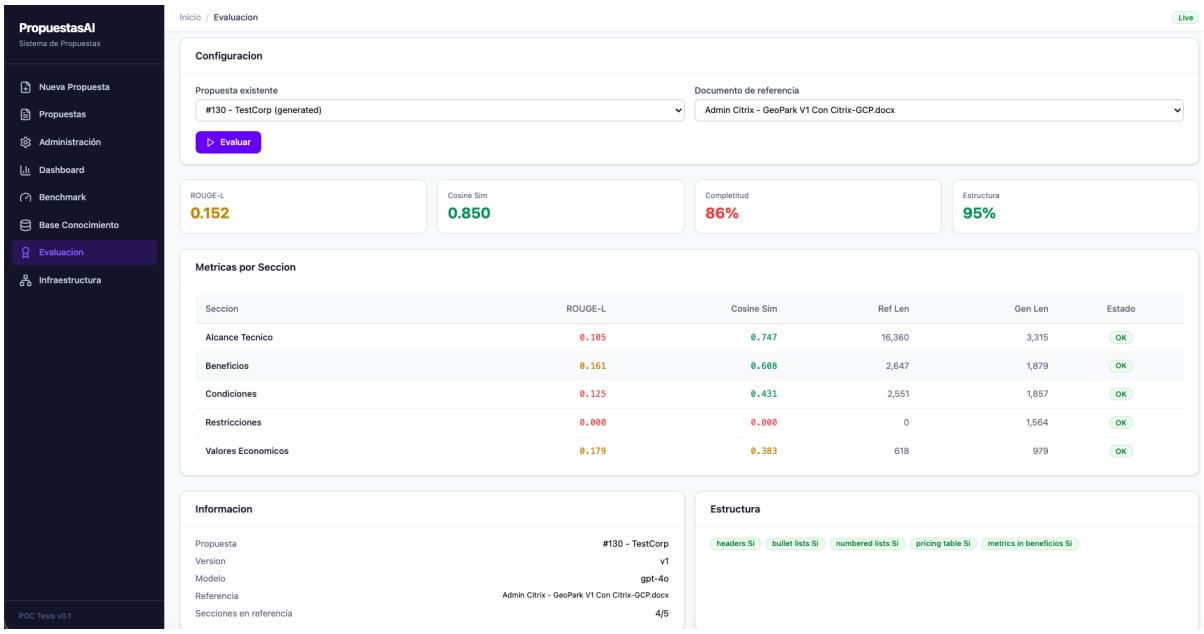


Figura 7.3: Panel de evaluaciones

7.3. Métricas operativas del Generative AI Hub

Además de la evaluación documental, la prueba de concepto permitió instrumentar métricas de observabilidad y comportamiento operativo.

Durante el periodo de ejecución se registraron:

- Latencia promedio global de 11.831 ms.
- Tasa de fallback de 26.7 %.
- Tasa de error global de 16.9 %.
- Costo acumulado de 9.4581 USD.

- Consumo total de 428.821 tokens.

Estas métricas permitieron validar el comportamiento integral del sistema bajo carga y múltiples proveedores.

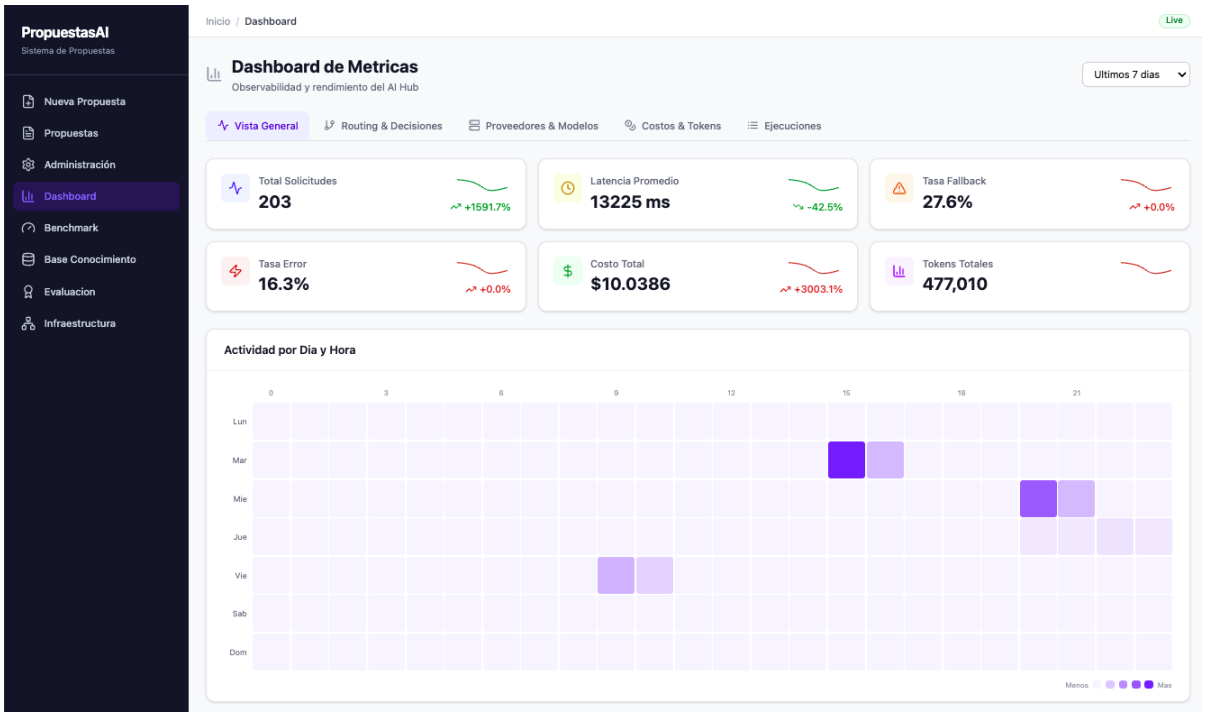


Figura 7.4: Dashboard de Métricas

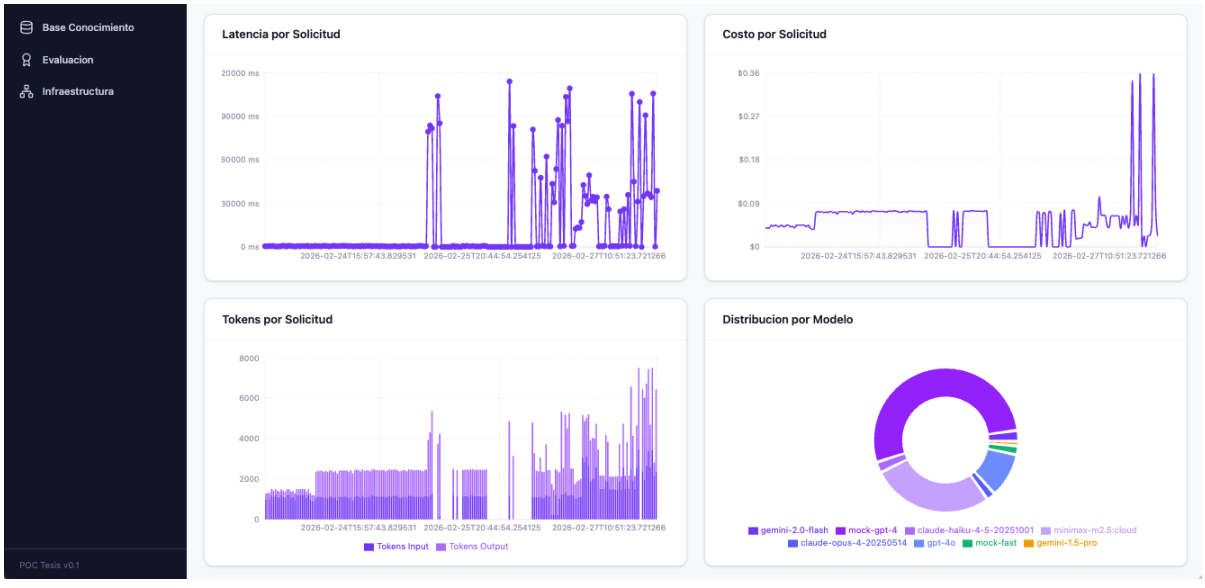


Figura 7.5: Dashboard de Métricas 2

7.3.1. Latencia por modelo (P50 y P95)

Se analizaron percentiles P50 y P95 por modelo. Los modelos de menor complejidad presentaron latencias inferiores a 1 segundo, mientras que modelos de mayor capacidad, como Claude Opus y MiniMax, alcanzaron latencias P95 superiores a 100.000 ms en algunos escenarios.

La diferencia entre P50 y P95 evidenció variabilidad bajo carga y justificó el uso de enrutamiento dinámico.

Comparacion Detallada de Modelos

MODELO	PROVEEDOR	REQUESTS	LATENCIA P50	LATENCIA P95	COSTO PROM	CALIDAD	ERROR %	EFICIENCIA TOKENS
mock-gpt-4	mock	111	481 ms	766 ms	$0.0653	0.70	0.0%	0.91
minimax-m2.5:cloud	ollama	58	81779 ms	109319 ms	$0.0051	0.75	0.0%	2.86
gpt-4o	openai	20	31860 ms	49390 ms	$0.0380	0.95	0.0%	0.88
claude-haiku-4-5-20251001	anthropic	6	36178 ms	38671 ms	$0.0233	0.78	0.0%	1.47
gemini-2.0-flash	gemini	4	697 ms	777 ms	$0.0738	0.85	0.0%	1.15
mock-fast	mock	3	684 ms	728 ms	$0.0706	0.50	0.0%	0.87
claude-opus-4-20250514	anthropic	3	105546 ms	105709 ms	$0.3539	0.98	0.0%	1.32
gemini-1.5-pro	unknown	1	653 ms	653 ms	$0.0759	0.00	0.0%	1.08

Figura 7.6: Métricas de latencia

7.3.2. Costo y eficiencia de tokens

Se evaluó la relación entre costo promedio por solicitud y calidad obtenida. GPT-4o presentó un equilibrio adecuado entre costo (0.0377 USD promedio) y calidad (0.95), mientras que Claude Opus alcanzó mayor calidad con mayor costo unitario (0.3515 USD promedio).

El análisis de eficiencia de tokens mostró diferencias significativas entre modelos, confirmando la pertinencia del enfoque multi-criterio definido en la arquitectura.

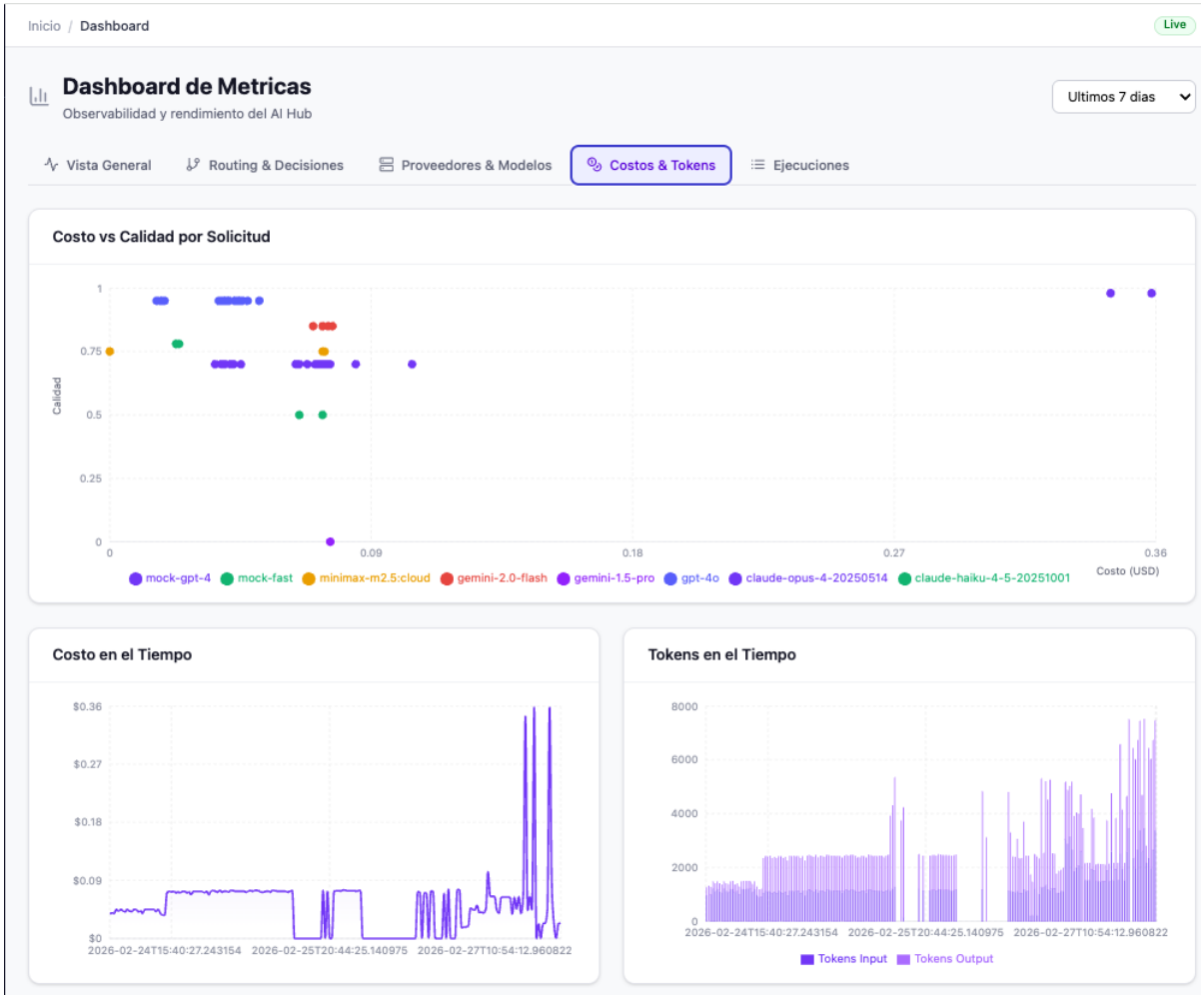


Figura 7.7: Métricas de tokens

Eficiencia de Tokens por Modelo

Modelo	Input Total	Output Total	Ratio	Costo/Token Output
gemini-2.0-flash	4,572	5,264	1.15x	$0.00005606
mock-gpt-4	126,304	115,268	0.91x	$0.00006287
claude-haiku-4-5-20251001	16,750	24,576	1.47x	$0.00000568
minimax-m2.5:cloud	24,564	70,271	2.86x	$0.00000419
claude-opus-4-20250514	9,334	12,288	1.32x	$0.00008639
gpt-4o	41,712	36,768	0.88x	$0.00002067
mock-fast	3,768	3,290	0.87x	$0.00006436
gemini-1.5-pro	1,214	1,316	1.08x	$0.00005767

Global: 228,218 input / 269,041 output = 1.18x eficiencia

Figura 7.8: Eficiencia de Tokens por Modelo

7.3.3. Distribución por proveedor

La distribución de solicitudes por proveedor muestra una concentración significativa en los modelos GPT-4o, seguida por otros modelos con menor participación relativa. Esta diferencia sugiere una preferencia operativa o una mayor frecuencia de asignación hacia dicho modelo dentro del sistema de enrutamiento.

En contraste, otros modelos presentan una participación más reducida, lo que indica un uso más específico o condicionado a ciertos tipos de consultas. La variación observada entre proveedores refleja el comportamiento dinámico del router, el cual asigna solicitudes en función de criterios de selección previamente definidos.

No se evidencia una distribución uniforme entre modelos, sino una tendencia de concentración en determinados proveedores, lo que podría estar relacionado con factores como disponibilidad, configuración del sistema o desempeño relativo en determinados tipos de tareas.

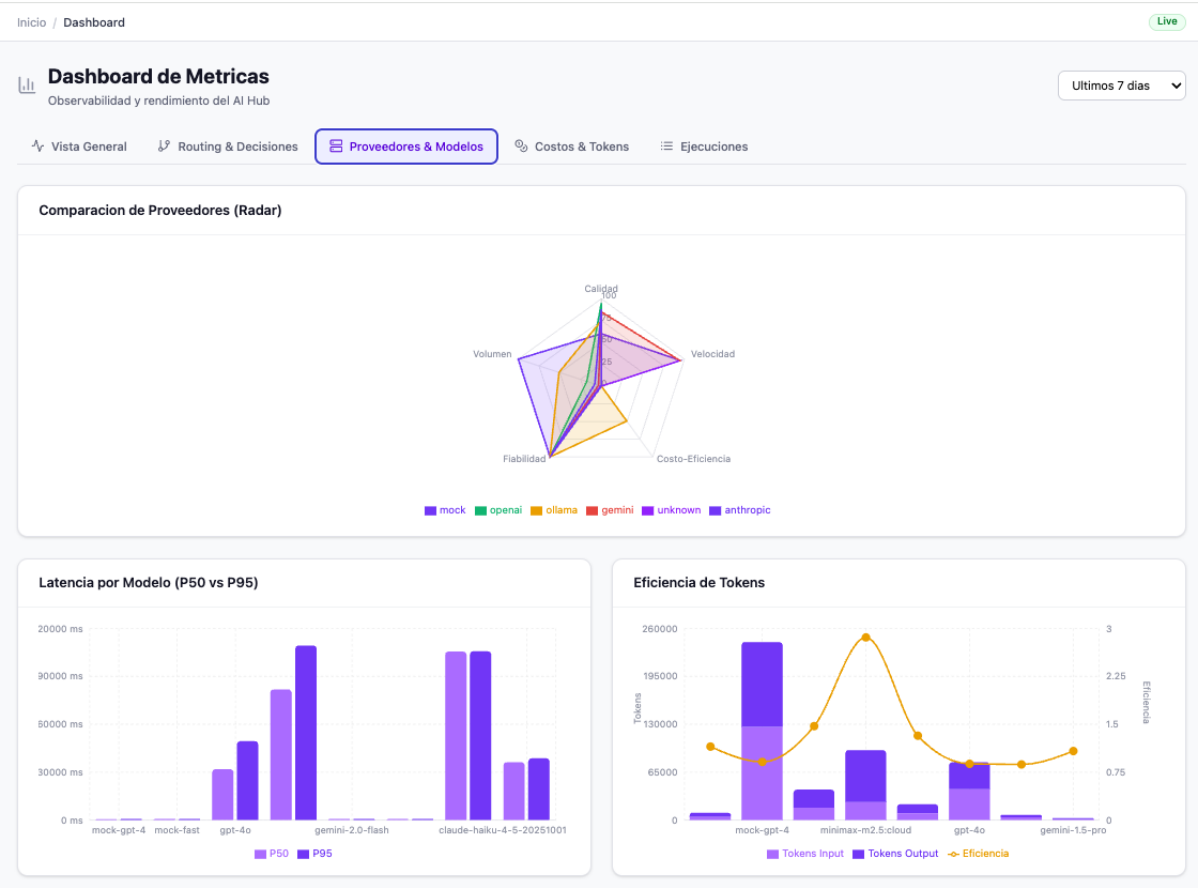


Figura 7.9: Distribución por proveedor

7.3.4. Comportamiento temporal

Los gráficos de costo en el tiempo y tokens en el tiempo mostraron picos asociados a ejecuciones con modelos de mayor capacidad. Estos resultados confirmaron la capacidad del sistema para capturar telemetría detallada y soportar gobernanza financiera.

7.4. Comparación entre modelos

El análisis comparativo evidenció perfiles diferenciados:

- GPT-4o presentó el mejor equilibrio entre tiempo de generación (31.6 s promedio) y completitud.
- Claude Opus obtuvo ligeramente mejor ROUGE-L, pero con mayor latencia (100 s promedio).
- MiniMax mostró alta eficiencia de tokens, pero presentó inestabilidad en un escenario, activándose fallback correctamente.

No existió un modelo universalmente superior; el desempeño dependió del criterio evaluado (velocidad, costo, similitud o completitud).

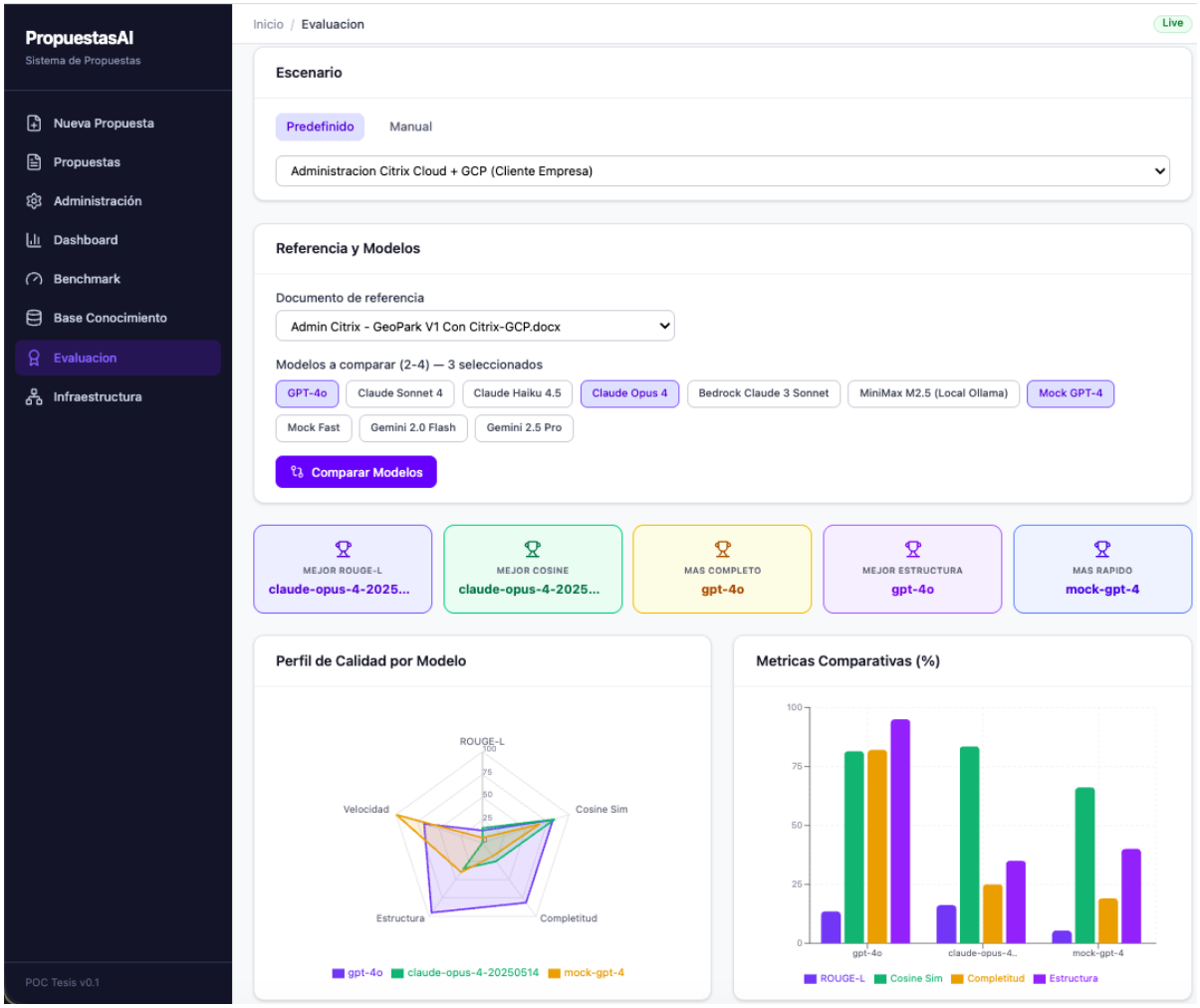


Figura 7.10: Distribución por modelo

Capítulo 8

Resultados Obtenidos

Los resultados se presentan en dos dimensiones principales: los derivados de la evaluación arquitectónica basada en escenarios y los obtenidos durante la validación técnica en entorno controlado.

8.1. Resultados de la evaluación arquitectónica

La aplicación del método SAAM permitió analizar la arquitectura propuesta frente a escenarios relacionados con seguridad, rendimiento y fiabilidad. A partir de la revisión sistemática de decisiones de diseño y su contraste con los escenarios formulados, se identificaron mecanismos explícitos que respondieron a los riesgos planteados.

En el ámbito de la seguridad, la centralización de la integración con proveedores externos mediante el Generative AI Hub permitió aislar la lógica de negocio de la invocación directa de modelos de inteligencia artificial. Esta separación facilitó la aplicación de controles de acceso, manejo de credenciales y trazabilidad de solicitudes.

En términos de rendimiento, se observó que la arquitectura incorporó desacoplamiento entre la interfaz de usuario y los procesos intensivos de generación. La introducción de procesamiento asíncrono en componentes críticos permitió reducir la dependencia de operaciones sincrónicas prolongadas.

Respecto a la fiabilidad, se documentó la presencia de mecanismos de manejo explícito de errores, reintentos controlados y enrutamiento alternativo ante fallos simulados de proveedores externos. La persistencia de estados y el almacenamiento de versiones contribuyeron a preservar la continuidad operativa ante escenarios de interrupción parcial.

Los escenarios evaluados mostraron correspondencia entre los atributos de calidad priorizados y los mecanismos arquitectónicos definidos.

8.2. Resultados de la prueba de concepto

La prueba de concepto permitió implementar y observar el comportamiento de los componentes principales de la arquitectura en un entorno controlado. Se desarrolló un módulo funcional que integró recuperación semántica de documentos históricos y generación asistida de contenido mediante modelos de inteligencia artificial.

Durante las pruebas realizadas, el sistema logró recuperar información relevante a partir de búsquedas basadas en similitud semántica, utilizando embeddings almacenados en un repositorio vectorial. Los documentos recuperados fueron incorporados como contexto para la generación de propuestas preliminares.

Asimismo, se verificó el funcionamiento del componente de enrutamiento dinámico, el cual permitió dirigir solicitudes hacia distintos modelos según criterios previamente definidos. Ante la simulación de indisponibilidad de un proveedor, se activaron mecanismos alternativos de respuesta, evidenciando el comportamiento esperado del diseño.

El despliegue en entorno local mediante contenedores permitió validar la integración entre los componentes de backend, almacenamiento y servicios de inteligencia artificial. Posteriormente, el despliegue en entorno cloud evidenció coherencia estructural con el diseño planteado en los diagramas arquitectónicos.

Si bien la prueba se ejecutó en condiciones controladas y con alcance limitado, permitió observar la viabilidad técnica de los elementos centrales del modelo propuesto.

8.3. Trazabilidad de resultados frente a los objetivos

Los resultados obtenidos se relacionaron con los objetivos planteados en el proyecto en términos de evidencia técnica y validación parcial. En particular, el diagnóstico del proceso actual aportó insumos para caracterizar las ineficiencias y oportunidades de automatización abordadas en el diseño. Asimismo, la arquitectura propuesta y su evaluación mediante escenarios permitieron observar mecanismos explícitos asociados a los atributos de calidad priorizados.

De manera complementaria, la prueba de concepto proporcionó evidencia funcional sobre la integración de recuperación aumentada (RAG), la incorporación de un hub de integración para desacoplar proveedores y la ejecución de un flujo básico de generación asistida. Esta trazabilidad se presenta como una relación de soporte técnico entre lo planteado y lo observado, sin constituir una validación operativa a escala productiva.

8.4. Observaciones derivadas de la validación

A partir de la evaluación arquitectónica y de los resultados obtenidos en la prueba de concepto, se pudo establecer que el diseño propuesto no se limitó a una formulación teórica, sino que demostró coherencia estructural y viabilidad técnica en su implementación parcial.

En el plano empírico, se procesaron 195 solicitudes a través del Generative AI Hub, incluyendo pruebas estructuradas de atributos de calidad, generación completa de propuestas comerciales y escenarios específicos para validar el enrutamiento dinámico y los mecanismos de recuperación. La ejecución sostenida de estos escenarios permitió observar un comportamiento estable del sistema, sin interrupciones críticas ni degradación funcional.

Desde la perspectiva arquitectónica, la validación no se redujo a comprobar que el sistema generara contenido, sino que examinó atributos como desacoplamiento, resiliencia, control y gobernanza multivendor. La activación automática del mecanismo de fallback ante la simulación de indisponibilidad de un proveedor evidenció que la tolerancia a fallos definida en el diseño fue efectivamente implementada y operativa.

Asimismo, la evaluación del enrutamiento dinámico mostró que las decisiones del sistema se alinearon con los criterios definidos en la arquitectura, priorizando calidad en tareas críticas y eficiencia en tareas exploratorias. Este comportamiento confirmó que la selección de modelos no fue arbitraria ni estática, sino gobernada por reglas explícitas.

La comparación entre el entorno Docker local y el despliegue en AWS reforzó esta conclusión al evidenciar consistencia funcional entre entornos, lo que confirmó la independencia de la lógica del Hub respecto a la infraestructura subyacente. La diferencia observada en escalabilidad respondió a capacidades propias del entorno cloud, no a limitaciones del diseño.

En conjunto, la evaluación permitió constatar que la arquitectura propuesta logró materializar los atributos de calidad priorizados —resiliencia, desacoplamiento y control multivendor— dentro del alcance definido para la prueba de concepto.

Capítulo 9

Conclusiones

9.1. Conclusiones

El presente trabajo tuvo como propósito diseñar una arquitectura de software orientada a automatizar la generación de ofertas comerciales mediante la integración de inteligencia artificial multivendor, bajo principios de calidad y escalabilidad en la nube, conforme al objetivo general planteado (cf. 1.3.1).

El análisis detallado del proceso actual de generación de propuestas permitió identificar ineficiencias estructurales asociadas a reprocesos, dispersión documental y dependencia del conocimiento individual, fundamentando la necesidad de una solución arquitectónica que habilitara reutilización sistemática y control del conocimiento (cf. **OE1:**).

A partir de este diagnóstico, se estructuró una arquitectura que incorpora mecanismos explícitos para soportar atributos de calidad como seguridad, rendimiento, fiabilidad y escalabilidad (cf. **OE2:**). La separación de responsabilidades, el desacoplamiento entre componentes y la asincronía en procesos intensivos materializan estos atributos a nivel de diseño. La evaluación mediante escenarios y el método SAAM permitió validar que dichas decisiones responden a riesgos reales de evolución tecnológica y crecimiento operativo, en coherencia con principios consolidados de arquitectura de software Bass et al. (2022); Richards and Ford (2020).

El diseño del Generative AI Hub como capa de orquestación central permitió integrar múltiples mecanismos de IA de manera desacoplada y gobernada por políticas transversales (cf. **OE3:**). Esta aproximación reduce el riesgo de dependencia tecnológica y facilita la aplicación uniforme de controles de seguridad, auditoría y observabilidad. Asimismo, la incorporación de recuperación contextual (RAG) contribuye a mejorar la confiabilidad del contenido generado, alineándose con enfoques recientes en la literatura Lewis et al. (2020).

La prueba de concepto desarrollada permitió validar parcialmente el comportamiento del sistema y su capacidad para reutilizar información histórica de propuestas comerciales (cf. **OE4:**). La ejecución en entorno local y en infraestructura cloud demostró la viabilidad técnica del modelo y su coherencia con principios de arquitectura cloud-native.

Finalmente, la evaluación arquitectónica basada en escenarios y SAAM permitió analizar la alineación del diseño con requisitos de calidad y sostenibilidad técnica en un contexto empresarial real (cf. **OE5:**). Esta validación refuerza la solidez del enfoque adoptado y su capacidad de adaptación ante cambios tecnológicos o de carga.

Desde una perspectiva académica, el trabajo contribuye con un modelo estructurado para integrar inteligencia artificial generativa en procesos empresariales críticos, combinando fundamentos clásicos de arquitectura de software con prácticas emergentes de gobernanza de IA. Desde una perspectiva organizacional, demuestra que la adopción de capacidades generativas puede realizarse de forma controlada, desacoplada y alineada con atributos de calidad, evitando integraciones ad hoc que comprometan sostenibilidad técnica.

No obstante, los resultados deben interpretarse considerando que la validación se realizó mediante una prueba de concepto controlada y no en un entorno productivo con carga sostenida. Asimismo, la evaluación de la calidad semántica del contenido generado no se abordó mediante métricas cuantitativas exhaustivas, lo cual constituye una oportunidad clara para profundización futura.

En síntesis, la arquitectura propuesta cumple el propósito planteado, integrando automatización generativa, evaluación arquitectónica formal y validación experimental dentro de un marco coherente, sostenible y alineado con los objetivos definidos para el proyecto.

9.2. Trabajos futuros

A partir de los resultados obtenidos en el desarrollo y validación del proyecto, se identifican oportunidades de evolución y expansión de la solución propuesta:

- Realizar pruebas de carga y desempeño en un entorno productivo controlado, incluyendo mediciones detalladas de tiempos de respuesta bajo alta concurrencia, considerando métricas de percentiles (por ejemplo, el tiempo dentro del cual se completan el 95 % o el 99 % de las solicitudes).
- Diseñar un esquema sistemático de evaluación de la calidad del contenido generado, incorporando métricas reproducibles y validación estructurada por expertos del dominio comercial, con el fin de convertir la calidad semántica en una variable medible y comparable.
- Evolucionar el enrutamiento dinámico hacia enfoques adaptativos que integren telemetría histórica y criterios comparativos de desempeño, fortaleciendo la capacidad de optimización progresiva del sistema.
- Profundizar en mecanismos de gobernanza y cumplimiento normativo, incluyendo clasificación automática de información sensible y políticas diferenciadas según contexto organizacional o regulatorio.
- Integrar completamente la solución con el ecosistema corporativo (CRM, gestor documental y flujos de aprobación), consolidando trazabilidad, versionado y control integral del ciclo de vida de las propuestas.
- Incorporar mecanismos de retroalimentación del usuario sobre el contenido generado, permitiendo que las correcciones y valoraciones de los especialistas de preventa alimenten el refinamiento progresivo de prompts, reglas de enrutamiento y la base de conocimiento.

Estas líneas de trabajo permitirían consolidar la transición desde una validación arquitectónica experimental hacia una adopción empresarial plenamente operativa y sostenible.

9.3. Lecciones aprendidas

El desarrollo del proyecto permitió identificar aprendizajes relevantes para el diseño e integración de inteligencia artificial generativa en entornos empresariales:

- La arquitectura constituye el principal mecanismo de gobernanza de la IA. Sin una capa explícita de control y orquestación, la integración de modelos puede volverse opaca, difícil de auditar y compleja de evolucionar.
- El desacoplamiento multivendor introduce flexibilidad estructural frente a un ecosistema de modelos en constante cambio y reduce el riesgo de dependencia tecnológica.
- La efectividad de sistemas generativos empresariales depende tanto del modelo como de la calidad y organización del conocimiento disponible para contextualización; por tanto, la gestión documental es un componente estratégico.
- La evaluación arquitectónica basada en escenarios permite anticipar riesgos antes de comprometer recursos en implementaciones completas, fortaleciendo la toma de decisiones técnicas fundamentadas.
- La prueba de concepto cumple un papel crítico como mecanismo de validación temprana, revelando consideraciones prácticas que no siempre son evidentes en los modelos conceptuales.

Estos aprendizajes refuerzan la importancia de abordar la integración de inteligencia artificial generativa desde una perspectiva arquitectónica estructurada y alineada con atributos de calidad, más allá de su adopción como capacidad tecnológica aislada.

Bibliografía

Ahmed, S. et al. (2025). Addressing Security Challenges in AI-Driven Cloud Platforms: Risks and Mitigation Strategies. *ResearchGate preprint*.

Alansari, A. et al. (2025). Large Language Models Hallucination: A Comprehensive Survey. *arXiv preprint arXiv:2510.06265*.

Alhosban, A., Pesingu, S., and Kalyanam, K. (2024). CVL: A Cloud Vendor Lock-In Prediction Framework. *Mathematics*, 12(3):387.

Amazon Web Services (2023). Aws well-architected framework.

Amazon Web Services (2024). Multi-llm routing strategies for generative ai applications on aws. Accessed: 2025-05-15.

Amazon Web Services (2025). Amazon bedrock: Multi-model foundation architecture.

Anthropic (2026). Claude Opus 4.6. https://anthropic.com. Accedido: abril 2026.

Artificial Analysis (2026). LLM Leaderboard: Intelligence Index. https://artificialanalysis.ai/leaderboards/models. Accedido: abril 2026.

Baldini, I., Castro, P., Chang, K., Cheng, P., Fink, S., Ishakian, V., Mitchell, N., Muthusamy, V., Rabbah, R., Suter, P., and Watson, P. (2017). Serverless computing: Current trends and open problems. *Research Advances in Cloud Computing*, pages 1–20.

Bass, L., Clements, P., and Kazman, R. (2012). *Software Architecture in Practice*. Addison-Wesley Professional, Boston, MA, 3rd edition.

Bass, L., Clements, P., and Kazman, R. (2022). *Software Architecture in Practice*. Addison-Wesley, 4th edition.

Beyer, B. et al. (2022). *Site Reliability Engineering*. O'Reilly Media.

Bilal, E. and Åström, O. (2024). Generative ai for sales: A study on the potential of retrieval augmented generation for response automation in the request for proposal process in telecommunications. *KTH Royal Institute of Technology*. Degree Project in Technology, Skolan för Elektroteknik och Datavetenskap.

Bloch, M., Blumberg, S., and Laartz, J. (2012). Delivering large-scale IT projects on time, on budget, and on value. *McKinsey Digital*.

Boppana, L., Bhadoria, M., and Kodali, R. K. (2024). An open-source rag architecture for llms. In *TENCON 2024 - 2024 IEEE Region 10 Conference (TENCON)*, pages 43–46.

Brown, S. (2017). *Software Architecture for Developers—Volume 2: Visualise, document and explore your software architecture*. Leanpub, Ebook Edition.

Chandy, K. M., Schulte, R., and Misra, W. (2010). *Event Processing: Designing IT Systems for Agile Companies*. McGraw-Hill Education, New York.

Cuellar, F. A. C. (2024). Framework para la integración de herramientas de inteligencia artificial en los productos de software para el área de seguridad y salud en el trabajo desarrollados por la corporación talentum. Master's thesis, Pontificia Universidad Javeriana Cali, Cali, Colombia.

Erl, T. (2005). *Service-Oriented Architecture: Concepts, Technology, and Design*. Prentice Hall, Upper Saddle River, NJ.

Fehling, C., Leymann, F., Retter, R., Schupeck, W., and Arbitter, P. (2014). *Cloud Computing Patterns: Fundamentals to Design, Build, and Manage Cloud Applications*. Springer, Vienna, Austria.

Fowler, M. (2003). *Patterns of Enterprise Application Architecture*. Addison-Wesley.

Francesco, P. D., Lago, P., and Malavolta, I. (2019). Architecting with microservices: A systematic mapping study. *Journal of Systems and Software*, 150:1–31.

Gao, L. et al. (2022). Evaluating large language models: A survey. *arXiv preprint arXiv:2303.18223*.

Gartner (2024). Enterprise ai platforms and generative ai architecture patterns.

Google Cloud (2023). Vector search overview.

Google Cloud (2024). Vertex ai rag engine overview.

Google DeepMind (2026). Gemini 3.1 Pro: A smarter model for your most complex tasks. https://blog.google/innovation-and-ai/models-and-research/gemini-models/gemini-3-1-pro/. Accedido: abril 2026.

Gorton, I. (2011). *Essential Software Architecture*. Springer, Berlin.

HatchWorks (2024). Ai orchestration unleashed: What, why, & how for 2025. Accessed: 2025-05-15.

Hugging Face (2024). Transformers and inference endpoints architecture guide. https://huggingface.co/docs.

Huyen, C. (2023). *AI Engineering: Building Applications with Foundation Models*. O'Reilly Media.

Inmon, W. H. (2005). *Building the Data Warehouse*. Wiley, Hoboken, NJ, 4th edition.

Jurafsky, D. and Martin, J. H. (2023). *Speech and Language Processing*. Prentice Hall. Draft version.

Kazman, R., Klein, M., Barbacci, M., Longstaff, T., Lipson, H., and Carriere, J. (1998). The Architecture Tradeoff Analysis Method. Technical Report CMU/SEI-98-TR-008, Software Engineering Institute, Carnegie Mellon University.

Kazman, R., Klein, M., and Clements, P. (2000). ATAM: Method for Architecture Evaluation. Technical Report CMU/SEI-2000-TR-004, Software Engineering Institute, Carnegie Mellon University.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., tau Yih, W., Rocktäschel, T., and Riedel, S. (2020). Retrieval-augmented generation for knowledge-intensive nlp tasks. *arXiv preprint arXiv:2005.11401*.

Lin, C.-Y. (2004). Rouge: A package for automatic evaluation of summaries. In *Proceedings of the Workshop on Text Summarization Branches Out*.

Liu, Y. (2023). Implications of generative artificial intelligence for the development of the media industry. *Shanghai Lida University*.

Lucidchart (2025). What is the pre-sales process? Accessed: 2025-02-26.

Medina, R. A. M. (2024). Arquitectura de software para vaova travel. Trabajo de grado de maestría, Pontificia Universidad Javeriana Cali, Santiago de Cali, Colombia. Disponible en línea.

Menlo Ventures (2024). 2024: The state of generative ai in the enterprise.

Microsoft (2024). Azure openai service documentation. Accessed: 2025-05-15.

Microsoft Azure Architecture Center (2023). Circuit breaker pattern.

Montoya, R. J. T. and Burbano, D. F. C. (2023). Plan de negocio para creación de turismo ia, empresa con enfoque en prestación. Technical report, Universidad Ean.

Muennighoff, N. et al. (2023). Mteb: Massive text embedding benchmark. *arXiv preprint arXiv:2210.07316*.

Newman, S. (2015). *Building Microservices*. O'Reilly Media.

Newman, S. (2022). *Building Microservices: Designing Fine-Grained Systems*. O'Reilly Media, 2 edition.

Nganga, A. (2025). How Great Engineers Make Architectural Decisions — ADRs, Trade-offs, and an ATAM-Lite Checklist. Azure Architecture Blog, Microsoft Tech Community. Accedido: abril 2026.

OpenAI (2023a). Embeddings guide.

OpenAI (2023b). Openai api documentation and architecture overview. https://platform.openai.com/docs.

OpenAI (2026). Introducing GPT-5.4. https://openai.com/index/introducing-gpt-5-4/. Accedido: abril 2026.

Paulose, R. and Neelanath, V. (2024). Generative ai-driven automation of business process reimagination. In *2024 IEEE Recent Advances in Intelligent Computational Systems (RAICS)*, pages 1–6.

Pinecone (2023). Vector database guide.

QorusDocs (2026). IT & SaaS Proposal Trends: 7 Insights from the 10th Annual Benchmark Report. https://www.qorusdocs.com/blog/it-saas-proposal-trends-7-insights-for-2026. Accedido: marzo 2026.

Reimers, N. and Gurevych, I. (2019). Sentence-bert: Sentence embeddings using siamese bert-networks. In *EMNLP*.

Richards, M. and Ford, N. (2020). *Fundamentals of Software Architecture: An Engineering Approach*. O'Reilly Media.

Salas, J., de Barros Vidal, F., and Martinez-Trinidad, F. (2019). Deep learning: Current state. *IEEE Latin America Transactions*, 17(12):1925–1945.

Seifi, N. and Chugh, M. (2025). Multi-LLM routing strategies for generative AI applications on AWS. AWS Machine Learning Blog. Accedido: marzo 2026.

Vázquez-Ingelmo, A., García-Holgado, A., and García-Peñalvo, F. J. (2020). C4 model in a software engineering subject to ease the comprehension of uml and the software. In *2020 IEEE Global Engineering Education Conference (EDUCON)*, pages 919–924.

Zambrano Barco, G. D. (2024). Plataforma tecnológica para habilitar la venta consultiva de un producto digital con inteligencia artificial. Trabajo de grado para optar al título de Magíster en Ingeniería de Software.

Zhang, T. et al. (2023). A survey on automatic text summarization evaluation. *ACM Computing Surveys*.

Zhang, W. et al. (2025). Mitigating Hallucination in Large Language Models: An Application-Oriented Survey on RAG, Reasoning, and Agentic Systems. *arXiv preprint arXiv:2510.24476*.

Zhao, M. et al. (2024). Retrieval-augmented generation: A survey. *ACM Computing Surveys*.