Universidad CENFOTEC

Maestría Profesional en Ingeniería del Software con énfasis en Inteligencia Artificial Aplicada

Tema:

Evaluación de la generación de casos de pruebas basados en requerimientos con modelos Large Language Model (LLM) para acelerar procesos en la industria aeroespacial.

Elaborado por:

Kendall Méndez Hidalgo

Fecha: Abril, 2026

Tabla de Contenido

Tabla de Figuras

Índice de Tablas

Abstract

Este trabajo final de graduación evalúa la efectividad de los Modelos de Lenguaje Grande (LLM) en la generación automatizada de casos de prueba basados en requerimientos para acelerar la verificación de software crítico en la industria aeroespacial. Mediante un enfoque alternativo, se comparó el desempeño cuantitativo y cualitativo de tres paradigmas de IA: RAG, *prompts* avanzados y agentes autónomos. Se contrasta el modelo comercial GPT-5 frente al de código abierto Mistral (7B).

Las evaluaciones emplearon un *benchmark* de cerca de 90 requerimientos simulados bajo la sintaxis estandarizada EARS, de los cuales 15 fueron utilizados activamente para validación y recolección de resultados. Los resultados cuantitativos demostraron una reducción drástica del esfuerzo inicial, disminuyendo el tiempo de generación de más de 50 minutos por requerimiento por la vía manual a menos de 1 minuto bajo el entorno asistido. A nivel de desempeño, la combinación de GPT-5 con *prompts* avanzados obtuvo la mayor precisión con un puntaje F1 promedio de 0.849 , mientras que la arquitectura agéntica basada en LangGraph con GPT-5 exhibió la mayor estabilidad y consistencia iterativa, alcanzando una sensibilidad promedio de 0.895 y reduciendo la omisión de pruebas gracias a la especialización de tareas en nodos acotados.

Por el contrario, el modelo Mistral evidenció notables deficiencias fuera de la lógica puramente booleana, mostrando una alta variabilidad y errores al procesar requerimientos con variables de rango continuo, tolerancias y operaciones matemáticas complejas.

Asimismo, el análisis cualitativo reveló vulnerabilidades inherentes a la naturaleza probabilística de los LLM, tales como alucinaciones estadísticas, omisión sistemática de condiciones de frontera y una marcada tendencia al sobretesteo (*over-testing*) que genera datos redundantes y exige depuración humana adicional. Como contribución científica y tecnológica fundamental, la investigación aporta una guía práctica de integración alineada rigurosamente con el estándar DO-178C y los

siete principios de la hoja de ruta de seguridad para IA de la Administración Federal de Aviación (FAA, 2024), además de un prototipo funcional interactivo desarrollado con LangChain, LangGraph y React que permite la interacción dinámica del usuario con modelos LLM para la generación de casos de prueba.

Finalmente, se concluye categóricamente que, debido al comportamiento no determinista de la IA generativa, las herramientas basadas en LLM no cumplen con los criterios de repetibilidad exigidos para la calificación de herramientas bajo la norma DO-330, lo que imposibilita su uso para reemplazar o automatizar de forma autónoma cualquier actividad de verificación. Por lo tanto, el diseño arquitectónico aeroespacial debe configurarse estrictamente bajo el paradigma de "humano en el flujo de trabajo" (*human-in-the-loop*). En este esquema, el sistema opera exclusivamente como un copiloto de ingeniería que acelera las propuestas iniciales, mientras que el profesional humano retiene de forma obligatoria la supervisión crítica, la validación de matrices de trazabilidad y la responsabilidad final de la seguridad del software de aviónica.

Palabras Clave: modelos *LLM*, casos de prueba basados en requerimientos, industria aeroespacial, DO-178C, verificación, software crítico, humano en el flujo de trabajo, estandarización, eficiencia.

1 Capítulo 1. Introducción

En este capítulo se introduce el tema de investigación, se explica el problema, el alcance, los objetivos y el estado de la cuestión. Las siguientes secciones detallan con mayor profundidad cada uno de estos temas.

1.1 Generalidades

El aumento de la complejidad de los sistemas aviónicos en la aviación civil requiere procesos rigurosos de aseguramiento y aprobación para garantizar el cumplimiento de las regulaciones federales. Para los sistemas basados en software, la Administración Federal de Aviación (FAA) reconoce la norma DO-178C como un medio aceptable de cumplimiento, tal como se indica en la Circular de Asesoramiento AC 20-115D.

La normal DO-178C establece las pruebas basadas en requerimientos como el método más efectivo para identificar errores. Este procedimiento consiste en analizar la descripción del requerimiento y a partir de esto, identificar casos de prueba que permitan verificar completamente el comportamiento y funcionalidad documentada en los requerimientos.

Dependiendo del requerimiento, es necesario incluir más o menos casos de prueba; por ejemplo, si el requerimiento describe funcionalidades relacionadas a máquinas de estado se deberán incluir en los casos de prueba todas las transiciones posibles de estado, así como las transiciones inválidas con el fin de demostrar que el sistema no permite dichos cambios.

La norma DO-178C especifica las pruebas mínimas que se esperan de acuerdo con el tipo de requerimiento.

Identificar casos de prueba es un procedimiento predominantemente manual en el cual los ingenieros analizan el tipo de requerimiento asignado y posteriormente generan los casos de prueba que se necesitan para verificar la correcta implementación del mismo.

Los casos de prueba (*test cases*) es el primer paso en el desarrollo de pruebas (*test procedure*) y, aunque no representa un gran porcentaje de la totalidad

del esfuerzo, es de gran importancia ya que es una de las bases de las pruebas como tal. Si la prueba no tiene los casos necesarios para verificar correctamente la implementación del requerimiento, se expone al riesgo de no encontrar errores en etapas tempranas del proyecto y, además, se aumenta el tiempo de corrección de pruebas en etapas de revisiones formales, ya que no se cumple con las pruebas mínimas requeridas según los planes del proyecto.

1.2 Antecedentes del Problema

En la industria aeroespacial, las pruebas basadas en requerimientos son el método sugerido por la guía DO-178C. Dichas pruebas comienzan con la identificación de casos de prueba, que es realizada manualmente, y representa aproximadamente un 10% del esfuerzo total del desarrollo de pruebas. La optimización de esta etapa podría traducirse en un ahorro considerable de tiempo y costos para la empresa, permitiendo reasignar recursos a otras tareas.

También, la calidad de las pruebas muchas veces está directamente relacionada a la experiencia del ingeniero, por lo que se requiere de supervisión y planes de seguimiento para las personas que no tienen tanto tiempo trabajando en este tipo de actividades.

Los errores en la identificación de casos de prueba tienen consecuencias directas: incrementan el tiempo y los costos de revisión, pueden conducir a retrabajos costosos, retrasos en la obtención de certificaciones regulatorias y, en el peor de los casos, podrían derivar en fallos funcionales no detectados durante las pruebas, comprometiendo la seguridad y el cumplimiento normativo.

1.3 Definición y Descripción del Problema

El problema central de este trabajo radica en investigar y evaluar la efectividad de los *LLM* en generar casos de prueba que permitan agilizar los procesos de verificación sin comprometer la calidad. Dado que este es un nuevo enfoque con pocos estudios en el tema, una de las dificultades principales es la limitada disponibilidad de datos que respalden buenas prácticas. Por ello, es necesario definir también una metodología para implementar y evaluar este tipo de tecnología.

Los modelos *LLM* ofrecen una solución potencial, al interpretar requerimientos en lenguaje natural y generar casos de prueba automáticamente. Sin embargo, su implementación enfrenta desafíos específicos:

- **Ambigüedad en requerimientos técnicos:** Los *LLM* pueden malinterpretar términos técnicos, generando casos de prueba que no cumplen con los estándares de trazabilidad.

- **Falta de estandarización:** No existen procedimientos consolidados para integrar *LLMs* en procesos de prueba aeroespaciales, lo que complica su adopción.

- **Preocupaciones éticas:** El uso de *LLMs* en sistemas críticos plantea riesgos, como la generación de casos de prueba incorrectos que podrían comprometer la seguridad aérea.

El enfoque alternativo se selecciona para evaluar métodos de *LLM* cuantitativamente (precisión, sensibilidad) y cualitativamente (aceptación del usuario), aprovechando las fortalezas de los métodos mixtos para abordar necesidades técnicas y contextuales, y específicamente, para mitigar la ambigüedad en requerimientos técnicos.

La investigación aplicada es adecuada para proponer una guía que permita incorporar este tipo de tecnologías y diseñar un prototipo funcional para la generación de casos de prueba, aunque su limitación radica en la necesidad de validación exhaustiva para garantizar el cumplimiento con DO-178C. Este enfoque asegura una solución integral adaptada a los requisitos operativos y regulatorios de la industria de la aviación.

1.4 Justificación

Este proyecto es relevante porque aborda una necesidad crítica en la industria aeroespacial: reducir el tiempo y los costos asociados con la generación de casos de prueba, manteniendo altos estándares de calidad. La generación de

pruebas mediante *LLMs* puede disminuir significativamente el tiempo de desarrollo de pruebas y mejorar la cobertura de requerimientos. Además, la estandarización de procesos mediante una herramienta innovadora permite cumplir con normativas como DO-178C, garantizando seguridad y eficiencia. Este Trabajo Final de Graduación (TFG) aporta innovación al integrar tecnologías de punta en un sector donde la fiabilidad es crucial, beneficiando a empresas, ingenieros y usuarios finales.

1.5 Viabilidad

A nivel personal, se cuenta con 9 años de experiencia trabajando en la industria de aviación, principalmente, suministrando servicios de verificación de software para ayudar a los clientes a obtener la certificación de sus productos.

Se ha trabajado con múltiples empresas y proyectos, obteniendo experiencia y exposición a una amplia variedad de requerimientos. La experiencia en servicios de verificación y entendimiento de la guía DO-178C respaldan la iniciativa de esta investigación. El criterio de experto permite evaluar los resultados obtenidos y definir una guía para la implementación de los LLMs en la generación de casos de prueba.

1.5.1 Punto de Vista Técnico. El equipo de investigación cuenta con conocimientos en inteligencia artificial, desarrollo de software adquiridos durante la formación en Ingeniería Eléctrica y la maestría de Inteligencia Artificial. Además, se tiene acceso a modelos *LLM* de código abierto (*Mistral*) y comerciales (*GPT-5*), así como a *frameworks* modernos como *LangChain* y *LangGraph*. La capacitación adicional en técnicas de *prompt engineering* se realiza mediante recursos en línea y documentación técnica, asegurando la competencia técnica para ejecutar el proyecto.

En cuanto al procesamiento computacional, la ejecución de los modelos de gran escala como GPT-4o y GPT-5 se realizará mediante el consumo de *endpoints* gestionados en Microsoft Azure. Por otra parte, los modelos de código abierto como Mistral que se emplearán corresponden a versiones de aproximadamente 7 mil millones de parámetros, lo que posibilita su ejecución en entornos con recursos computacionales moderados sin requerir infraestructura especializada de alto rendimiento.

Esta decisión permite establecer una comparación sistemática entre modelos de distinta escala con el fin de analizar su impacto en los paradigmas de aplicación y en los tipos de tareas requeridas para la identificación de casos de prueba.

1.5.2 Punto de Vista Operativo. El proyecto se lleva a cabo sin interrumpir las operaciones de la empresa aeroespacial. Las pruebas piloto se realizan en un entorno controlado, utilizando datos simulados de requerimientos. La colaboración con ingenieros de pruebas se coordina mediante reuniones virtuales, minimizando la necesidad de recursos físicos.

Los requerimientos a utilizar son ficticios por lo que no hay riesgo de utilizar propiedad privada de los clientes. Dichos requerimientos fueron generados tomando en cuenta la variación en formato y tipos de requerimientos observados en otros proyectos.

1.5.3 Punto de Vista Económico. El costo teórico del proyecto se detalla en la Tabla 1. La mayor parte del costo corresponde a las horas de consultoría del investigador, asumidas por el propio estudiante. Los costos asociados a la colaboración de ingenieros de la empresa son mínimos, ya que se limitan a horas de revisión y validación. Adicional a las horas de investigador y consultoría, se debe pagar un costo por utilizar los *API* de *OpenAI* dado que no son gratuitos. El investigador asume dichos costos.

Tabla 1: Costos de la investigación

Rubro	**Costo**
De parte del investigador	
Horas de consultor	$24,000
Capacitación metodológica	$0 (recursos en línea)
Literatura	$200
Transportes	$0 (trabajo remoto)
Licencias de software	$300
Equipo computacional	$2000 (equipo personal)

1.6 Objetivos

Los objetivos se redactan utilizando la taxonomía de Bloom revisada, que clasifica los procesos cognitivos en niveles como recordar, comprender, aplicar, analizar, evaluar y crear. Esta taxonomía se selecciona por su claridad para estructurar objetivos desde lo básico hasta lo complejo, asegurando que el proyecto progrese de manera lógica hacia la solución del problema.

1.6.1 Objetivo General. Evaluar la generación de casos de pruebas basados en requerimientos con modelos *Large Language Model* (*LLM*) para acelerar procesos de verificación en la industria aeroespacial.

- 1.6.2 Objetivos Específicos.
- Identificar las limitaciones y desafíos sobre el uso de modelos *LLM* para la generación de casos de prueba de software crítico en la industria aeroespacial.
- Describir el proceso de generación de casos de prueba recomendado por el estándar DO-178C con el fin de establecer una base metodológica para el uso de modelos LLM en la generación de casos de pruebas.
- Clasificar los requerimientos funcionales en categorías según el tipo de entradas y variables, con el fin de establecer una estrategia sistemática de generación de casos de prueba que facilite la evaluación posterior mediante modelos LLM.
- Analizar y comparar, a partir de la literatura existente, diferentes paradigmas de aplicación de los *LLM* para la generación de casos de prueba, evaluando su aplicabilidad a los requerimientos de la industria aeroespacial.
- Diseñar y desarrollar un prototipo funcional y una guía práctica para la integración de modelos *LLM* en la generación de casos de prueba, considerando las normativas DO-178C, aspectos éticos y de seguridad.

- Elaborar una guía de generación de casos de prueba según el tipo de requerimiento analizado que sirva de base para el entrenamiento del personal y desarrollo de herramientas impulsadas por *LLM* en la industria aeroespacial.

1.7 Alcances y Limitaciones

El presente trabajo utiliza datos simulados para garantizar la confidencialidad y reproducibilidad del estudio. Se dispone de ayuda por parte de la empresa Avionyx para la validación de los resultados. La guía y el prototipo están orientados a su contexto operativo y normativo, pero los datos de prueba y requerimientos utilizados no corresponden a proyectos o clientes reales.

1.7.1 Alcances.

- Un documento escrito que detalla el marco conceptual para integrar modelos *LLM* en la generación de casos de prueba para sistemas aeroespaciales, incluyendo una revisión sistemática de literatura.

- Una guía práctica para ingenieros de pruebas en la empresa Avionyx, describiendo cómo implementar *LLMs* en procesos de prueba cumpliendo con la normativa DO-178C.

- Un prototipo funcional de una herramienta basada en *LangChain*, Langgraph y *React* que genera casos de prueba automáticamente a partir de requerimientos en lenguaje natural.
- Un informe técnico con los resultados de la evaluación comparativa de métodos de uso de *LLMs*, incluyendo métricas de precisión y exactitud.

1.7.2 Limitaciones.

El proyecto no abarca las siguientes áreas:

- Desarrollo de un sistema completo listo para producción, ya que el enfoque es un prototipo de prueba de concepto.

- Validación de los casos de prueba generados en hardware real o entornos operativos, debido a la naturaleza simulada de los datos.

- Integración de la herramienta en los sistemas existentes, ya que esto requiere recursos adicionales fuera del alcance del TFG.

- Evaluación de *LLMs* en dominios no aeroespaciales, enfocándose exclusivamente en sistemas críticos de aeronaves.

1.8 Marco de Referencia Organizacional y Socioeconómico

En este apartado se incluye la historia, tipo de negocio, misión, visión, valores de la empresa que respaldan el estudio y algunas políticas de la empresa que se deben tomar en cuenta para este trabajo de la empresa Avionyx donde se estará implementando la prueba de concepto.

1.8.1 Historia**.** Fundada en 1989, Avionyx inició sus operaciones en Florida. Tras varios años de crecimiento constante, la empresa se expandió con la apertura de nuevas oficinas en Costa Rica. Impulsada por el desarrollo acelerado y el sólido apoyo de la comunidad local, todas las operaciones se trasladaron en 2008 a la Zona Franca de América. Desde sus inicios, Avionyx se ha especializado en el desarrollo y la verificación de software y firmware de aviónica, cumpliendo rigurosos estándares de la industria como DO-178C, DO-278A y DO-254. La empresa ha apoyado a numerosos fabricantes de aviónica, ofreciendo servicios que abarcan todo el ciclo de vida del desarrollo de software, incluyendo el análisis de requisitos, el diseño, la verificación y el soporte para la certificación. En 2022, Avionyx fue adquirida por Joby Aviation, empresa californiana dedicada al desarrollo de aeronaves eléctricas de despegue y aterrizaje vertical (*eVTOL*).

1.8.2 Tipo de Negocio y Mercado Meta**.** Avionyx es una empresa especializada en servicios de desarrollo y verificación de software en la industria aeroespacial. Los clientes son empresas responsables del desarrollo de equipos de aviación que necesitan ayuda para certificar sus productos de software. Avionyx brinda todo tipo de soporte necesario para desarrollar software con base en la guía DO-178C.

1.8.3 Misión, Visión y Valores**.** La misión de Avionyx se define como:
"Avionyx es una empresa de servicios de ingeniería que combina estrictos procesos de calidad, el personal más capacitado y soluciones innovadoras para desarrollar y dar soporte a software, hardware y sistemas de seguridad y misión crítica para la industria aeroespacial. Avionyx se esfuerza por maximizar el valor para sus clientes y empleados mediante el compromiso, el respeto, la integridad, el profesionalismo y el trabajo en equipo para lograr la satisfacción general y relaciones duraderas."

La visión de la compañía es:
"Aprovechar nuestra experiencia en servicios de ingeniería para ayudar a nuestros clientes a maximizar la calidad y el valor al ofrecer los productos de aviación más seguros, confiables y sostenibles del mundo."

Los valores de la empresa son la calidad del servicio, el compromiso, el respeto, la innovación, el profesionalismo y el trabajo en equipo.

1.8.4 Políticas Institucionales. Los nombres de los clientes y proyectos no se pueden mencionar. Además de las restricciones mencionadas, Avionyx mantiene un conjunto de políticas corporativas alineadas con sus prácticas de ingeniería y cumplimiento normativo:

- Protección de la confidencialidad: no está permitido utilizar información de clientes (requerimientos, especificaciones, documentación) salvo en entornos autorizados y con *NDA* vigentes.
- Cumplimiento normativo y certificatorio: todas las actividades de verificación y desarrollo de software deben adherirse a estándares DO-178C/DO-254, manteniendo trazabilidad completa desde requisitos hasta evidencia de verificación.
- Asistencia a auditorías: los equipos deben estar preparados para presentar evidencia de cumplimiento en auditorías internas y externas (*FAA*, *EASA*).
- Gestión de herramientas calificadas: se exige el uso de herramientas certificadas o calificadas según DO-330, y cualquier herramienta nueva que reemplace al humano en actividades de verificación debe seguir un proceso formal de evaluación y validación.

1.9 Revisión de literatura

1.9.1 Revisión sistemática

La revisión sistemática de literatura se realiza siguiendo la plantilla propuesta por Biolchini, Gomes, Cruz y Horta (2005), que estructura el proceso en preguntas de investigación, fuentes, criterios de selección y análisis.

- **Pregunta de investigación:** ¿Cuáles son los desafíos, mejores prácticas y resultados reportados en la literatura sobre el uso de modelos *LLM* para la generación de casos de prueba en software crítico, particularmente en la industria aeroespacial?
- **Fuentes:** Bases de datos académicas como I*EEE Xplore, ACM Digital Library, Springer y Google Scholar*, complementadas con repositorios de preprints como *arXiv* para estudios recientes.
- **Criterios de inclusión:** Artículos publicados entre 2022 y 2025, en inglés, que aborden *LLMs*, generación de casos de prueba y/o software crítico, priorizando estudios con resultados empíricos o aplicaciones en industrias reguladas (e.g., aeroespacial, médica).
- **Criterios de exclusión:** Artículos sin datos empíricos, centrados en dominios no críticos o que no mencionen *LLMs* explícitamente.
- **Proceso:** Se empleó una búsqueda avanzada en las fuentes académicas mencionadas anteriormente donde se filtró resultados por medio de las siguientes palabras clave: "*verification", "LLM", "Test case", "aerospace", "Artificial Intelligence", "requirement based test".* Estos se analizarán en detalle para extraer desafíos, métodos y resultados.

1.9.2 Estado de la cuestión

Dada la capacidad de los LLM para procesar texto, seguir instrucciones y extraer información del lenguaje natural, estos se han investigado como una herramienta para generar casos de prueba a partir de requerimientos textuales en diferentes industrias.

Trabajos previos han demostrado que los LLM son competentes en la generación de texto, la comprensión del lenguaje, el razonamiento aritmético y lógico, así como en la comprensión contextual (Wang, et al., 2024).

Korraprolu, Pinninti, y Reddy (2025) examinaron las capacidades de varios modelos para generar casos de prueba utilizando únicamente requerimientos textuales. En esta investigación, se empleó un solo prompt con diferentes modelos (BARD, ChatGPT3.5, Claude, Gemini, ChatGPT4.0 y Llama3) para evaluar los casos de prueba generados por cada uno.

El método de evaluación consistió en generar casos de prueba utilizando LLMs y luego compararlos con los obtenidos mediante Simulink, una herramienta comercial. La calidad de los casos de prueba se evaluó principalmente a través de la cobertura de código, analizando las porciones del código ejecutadas por los casos de prueba generados.

Además, los requerimientos se clasificaron en diferentes categorías con el fin de investigar la relación entre el tipo de requerimiento y la calidad de los casos de prueba correspondientes. Dado que la cobertura se utilizó como la métrica objetiva principal para evaluar la calidad de las pruebas, estas clasificaciones de requerimientos se derivaron con base en la complejidad ciclomática del código asociado a cada requerimiento.

Los resultados de este experimento mostraron que era posible generar casos de prueba utilizando LLMs; sin embargo, el porcentaje de cobertura disminuyó conforme aumentaba la complejidad de los requerimientos, lo que indica que los LLMs presentaban dificultades para comprender estructuras lógicas complejas dentro de los requerimientos.

Los hallazgos también revelaron que los LLMs tenían problemas para capturar relaciones entre requerimientos, lo cual limitaba su capacidad para generar casos de prueba con cobertura completa. Además, los requerimientos que involucran múltiples entradas, iteraciones o restricciones de tiempo fueron excluidos del alcance de este estudio.

Cabe destacar que en este estudio realizado por Korraprolu, Pinninti, y Reddy (2025), no se utilizaron múltiples *prompts* para analizar la variabilidad de respuestas. El estudio se limitó a utilizar un único prompt con informarción general acerca de la tarea a realizar y se probó con varios LLMs con el fin de observar las respuestas obtenidas con diferentes modelos.

Los LLMs también han sido utilizados por la industria automotriz para generar casos de prueba a partir de requerimientos textuales. El estudio realizado por Malmberg (2025) propone utilizar GPT-4.0 para generar casos de prueba a partir de los requerimientos incluidos en un documento.

Los documentos se clasifican en tres categorías dependiendo del nivel de complejidad de los requerimientos que contienen. Se consideraron de baja complejidad aquellos documentos asociados con menos de cinco casos de prueba esperados. Aquellos con seis a dieciséis casos de prueba esperados se clasificaron como de complejidad media, mientras que los documentos con más de dieciséis casos de prueba esperados se categorizaron como de alta complejidad.

En este estudio se utilizó un solo prompt, y los casos de prueba generados se analizaron en términos de cobertura de requerimientos, completitud, corrección y precisión (identificar señales relevantes sin detalles innecesarios).

Los resultados mostraron que el LLM fue capaz de generar pruebas utilizables en el 63 % de los casos evaluados para requerimientos más simples. Consistente con los hallazgos reportados por Korraprolu et al. (2025), la calidad de las pruebas disminuyó a medida que aumentaba la complejidad de los requerimientos. El estudio también confirmó que los LLMs pueden acelerar significativamente el proceso de generación de pruebas, reduciendo el tiempo requerido en un factor de 45 a 180. Sin embargo, debido a limitaciones como alucinaciones, complejidad de los requerimientos y los estrictos estándares de la industria, la supervisión humana sigue siendo necesaria para garantizar que los casos de prueba generados cumplan con los requisitos de la industria.

Aunque la mayoría de los requerimientos textuales describen la lógica del sistema y el comportamiento esperado en lenguaje natural, ciertos requerimientos

dependen de diagramas para representar la lógica de máquinas de estado y procesos secuenciales. Para abordar esto, se propuso un método que utiliza LLMs para generar casos de prueba a partir de diagramas de máquinas de estado en la industria aeroespacial (Su, Q., et al., 2024).

En este método se siguieron cinco pasos:

1. Extraer criterios de seguridad de los estándares de aviación (por ejemplo, DO-178C, ARP-4754A).
2. Extender las máquinas de estado con los criterios de seguridad utilizando LLMs.
3. Generar una nueva máquina de estado con los criterios de seguridad incorporados.
4. Realizar un recorrido de profundidad primero por los bordes del diagrama extendido para crear rutas de prueba.
5. Usar LLMs para optimizar los hiper parámetros y los procesos de mutación en algoritmos genéticos.

El método propuesto automatiza de manera efectiva la extensión de los diagramas de estado de seguridad y la generación de casos de prueba, reduciendo la dependencia de la experiencia del *tester* y mejorando la calidad de las pruebas. Además, redujo en un 90 % el número máximo de iteraciones al utilizar el algoritmo genético.

Considerando que los LLMs también han sido entrenados con datos de código y lenguajes de programación, distintos estudios han utilizado esta característica para evaluar el desempeño de los LLMs en la generación de casos de prueba para pruebas unitarias.

Li y Yuan (2024) llevaron a cabo un experimento para evaluar los casos de prueba generados a partir de la descripción de una función, en el cual se emplearon varios LLMs (StarChat, CodeLLama, GPT-3.5, GPT-4). En este estudio, se utilizaron dos conjuntos de datos de preguntas y respuestas de código para evaluar la corrección de las respuestas generadas por los LLMs. Los resultados indicaron que, aunque los modelos producían respuestas variadas para ejemplos simples, los

modelos más avanzados (por ejemplo, GPT-3.5 y GPT-4) generaban casos de prueba adicionales para escenarios complejos.

Se propuso un método alternativo basado en el *prompting* ReAct a través del *framework* TestChain, que emplea un LLM para la identificación de entradas de prueba y otro para la predicción de salidas. Al dividir las tareas entre modelos, este enfoque introdujo especialización en los distintos módulos del LLM. Los resultados mostraron que aplicar TestChain con GPT-4 mejoró el desempeño en un 13,84 % en comparación con el uso de GPT-4 de manera aislada sobre el mismo conjunto de datos (Li K. Y., 2024).

1.10 Métricas de Evaluación

Las métricas que se emplearán para la evaluación son la precisión, la sensibilidad y la puntuación F1.

$$Precisión = \frac{TP}{TP + FP} \quad (1)$$

$$Sensibilidad = \frac{TP}{TP + FN} \quad (2)$$

$$Puntuación\ F1 = 2 \times \left(\frac{Precisión \times Sensibilidad}{Precisión + Sensibilidad}\right) \quad (3)$$

Las ecuaciones (1), (2) y (3) muestran la implementación de cada métrica en donde *TP* representa el número de casos de prueba correctos, *FP* el número de casos de prueba sugeridos por el *LLM* que son incorrectos y *FN* el número de casos de prueba faltantes.

Los casos de prueba sugeridos por los *LLM* serán analizados utilizando dichas métricas para determinar cuál paradigma de implementación y *LLM* obtuvo mejores resultados, y para cuantificar los casos faltantes e incorrectos.

1.11 Diseño del Estudio de Caso

Para llevar a cabo esta evaluación, es necesario definir un *benchmark* de requerimientos con sus casos de prueba esperados correspondientes. Las descripciones de los requerimientos pueden incluir operadores booleanos, múltiples condiciones, algoritmos, funcionalidades de temporización, etc.

Se proponen dos categorías para la clasificación de requerimientos:

- Requerimientos con condiciones: Incluyen entradas y condiciones bajo las cuáles se deben implementar ciertas acciones.
- Requerimientos sin condiciones: el sistema debe implementar las acciones documentadas independientemente de condiciones o estado de las entradas.

También se requiere una lista de verificación para establecer los criterios que se seguirán al evaluar las respuestas de los LLM basándose en la guía DO-178C. La Tabla **2** muestra la lista que se utilizará para la verificación de casos de prueba sugeridos por los *LLM.*

Tabla 2: Criterios de evaluación de los casos de prueba

Criterio ID	Criterio de evaluación
CI-1	Los casos de prueba de verificación incluyen representaciones de clases de equivalencia para las entradas.
CI-2	Los casos de prueba verifican correctamente los límites.
CI-3	Los casos de prueba verifican correctamente los valores mínimos y máximos del rango.
CI-4	Los casos de prueba toman en consideración aspectos relacionados con variables que involucran tiempo.
CI-5	Los casos de prueba cubren operaciones Booleanas.
CI-6	Los casos de prueba incluyen el resultado esperado.
CI-7	Cuando es posible los casos de prueba incluyen representaciones de clases de equivalencia para entradas fuera de los límites.

CI-8	Los casos de prueba predicen correctamente los resultados esperados basándose en la lógica descrita en el requerimiento y en los valores de entrada especificados en cada caso de prueba.

Se utilizan tres métricas cuantitativas—precisión, puntuación F1 y sensibilidad—para analizar las respuestas de los LLM. Para los casos de prueba faltantes o incorrectos, se documentan los tipos de errores identificados con el fin de proporcionar una evaluación cualitativa del rendimiento del LLM.

Para llevar a cabo esta evaluación, se consideran dos experimentos diferentes. El primero consiste en evaluar las distintas combinaciones de LLMs y paradigmas de aplicación utilizando los requerimientos de para comparar los resultados entre los diferentes enfoques y modelos. El segundo es un experimento similar, pero esta vez añadiendo otra variable: las respuestas a las mismas preguntas por parte de ingenieros de verificación con diferentes niveles de experiencia en la industria aeroespacial.

El *benchmark* de requisitos se dividirá en dos grupos: uno para la comparación entre LLMs y otro para la comparación entre LLMs y humanos. El número de requerimientos utilizado para la comparación LLM–humano es significativamente menor que el utilizado para la comparación sólo entre LLMs, ya que la evaluación implica administrar una prueba a los participantes. Aunque el tipo y nivel de complejidad de los requisitos serán equivalentes en ambos experimentos, los requisitos específicos empleados serán diferentes.

2 Capítulo 2. Marco Teórico o Conceptual

Dado el enfoque alternativo de este trabajo, se opta por un marco conceptual, que consiste en la definición y clarificación de los principales conceptos que estructuran la investigación sobre la integración de modelos de lenguaje grande (*LLMs*) en la generación de casos de prueba para software crítico en la industria aeroespacial.

2.1 Metodología para la elaboración del marco conceptual

Siguiendo las recomendaciones metodológicas de la Universidad Cenfotec, este marco conceptual se elabora a partir de:

- **Revisión sistemática de literatura:** Se identificaron los conceptos clave relacionados con *LLMs*, pruebas basadas en requerimientos, verificación, DO-178C, y paradigmas de aplicación de los *LLMs*.

- **Nube y jerarquización de conceptos:** Se construyó una nube de conceptos y un mapa conceptual jerárquico para organizar los términos desde los más generales hasta los más específicos.

- **Definición top-down:** Cada concepto se presenta y desarrolla de lo general a lo particular, asegurando coherencia y fluidez en la exposición.

- **Criterios de fuentes:** Las definiciones y discusiones de cada concepto provienen de literatura académica y normativas reconocidas, conforme a criterios de rigor científico y autoridad.

2.2 Conceptos Fundamentales

La DO-178C es la norma internacional que regula el desarrollo y la certificación de software para sistemas aeronáuticos. Exige procesos rigurosos de verificación y validación, asegurando la trazabilidad entre requerimientos, código y pruebas. Cumplir con DO-178C es obligatorio para obtener certificaciones como la de la *FAA* o *EASA* (Solutions, 2025).

La prueba basada en requerimientos es el enfoque principal en industrias reguladas, ya que garantiza que cada requerimiento del sistema sea verificado a través de uno o más casos de prueba específicos. Según la guía DO-178C, este

método es esencial para lograr trazabilidad y cobertura, aspectos críticos en sistemas aeroespaciales (RTCA, 2011).

Las pruebas basadas en requerimientos son el método preferido por DO-178C ya que se ha demostrado ser el más efectivo para la detección de errores (RTCA, 2011). Este proceso comienza con un fase previa que es la generación de casos de prueba. Este proceso consiste en identificar los escenarios que van a ejecutarse en las pruebas basado en un análisis de los requerimientos. Los casos de prueba son un resumen de dichos escenarios y usualmente describen información de los valores de entrada de los requerimientos y las salidas esperadas en el sistema basado en la lógica y funcionalidad del requerimiento analizado. La guía DO-178C establece los criterios a tomar en cuenta para seleccionar la cantidad mínima de casos de prueba.

Esta fase inicial es una tarea repetitiva, laboriosa y depende en gran medida de la experiencia y práctica del ingeniero de verificación. Por esta razón, se motiva a analizar la efectividad de herramientas que asistan al ingeniero en esta tarea.

Los *Large Language Models (LLMs)* son modelos de inteligencia artificial diseñados para la arquitectura de *transformers* que les permite predecir la siguiente palabra con mayor probabilidad de aparición en una secuencia de datos. Estos modelos son entrenados con grandes volúmenes de texto, capaces de comprender y generar lenguaje natural de manera autónoma. Se han utilizado para tareas de traducción de lenguajes, resumen y análisis de texto, y conversaciones (Geroimenko, 2025).

Debido a que la gran mayoría de requerimientos son representados de manera textual, los *LLM* tienen potencial de asistir a los ingenieros de verificación en este tipo de tareas.

Existen diferentes métodos para emplear los *LLM* en el desarrollo de tareas requeridas por el usuario. Estos métodos se presentan en este trabajo como paradigmas de aplicación. En esta investigación, se analizarán tres métodos: *prompts, RAG y* agentes.

Los prompts son las entradas e instrucciones utilizadas para dirigir a los *LLM* a realizar las tareas solicitadas por el usuario. Existen buenas prácticas y técnicas de prompt engineering que pueden emplearse para mejorar la calidad de las respuestas generadas por los *LLM (*Geroimenko, 2025*)*.

RAG es un marco de trabajo para LLM que combina un método de recuperación con un generador de secuencia a secuencia que utiliza los *LLM* para predecir la siguiente palabra de la secuencia (Oche et al., 2025). Este método permite que el *LLM* amplíe su base de conocimientos hacia datos para los cuales no fue entrenado. Este enfoque reduce las alucinaciones y mejora la precisión y la relevancia de las respuestas de los *LLM* dentro de dominios específicos del conocimiento.

Los agentes LLM tienen la capacidad de percibir su entorno y tomar decisiones. Los agentes poseen características que les permiten manejar solicitudes más complejas y tareas de resolución de problemas durante el proceso de aplicación. Un agente tiene características individuales que provienen del conocimiento que se le proporcionó al modelo, de sus roles y de sus objetivos. Recibe información de herramientas o fuentes externas para comprender el entorno que lo rodea. Además, posee la capacidad de planificar acciones basadas en su percepción del entorno y su estado interno, lo que le permite ejecutar tareas que contribuyen a alcanzar sus objetivos (As Li et al., 2024).

Una limitación y problema de los *LLM* es que, en algunos casos, generan información incorrecta. Este fenómeno se llama alucinación y ocurre porque los *LLM* predicen palabras basándose en patrones que aprendieron durante su entrenamiento. Si el usuario realiza una pregunta que el modelo nunca ha visto antes, hará una conjetura estadística basada en los patrones observados durante su entrenamiento. Este proceso puede llevar a la producción de información inexacta que parece coherente y lógicamente consistente, a pesar de ser incorrecta (Coursaris et al., 2025).

2.3 Conclusión del Marco Conceptual

Este marco conceptual proporciona las bases necesarias para entender los desafíos, oportunidades y consideraciones éticas en la aplicación de *LLMs* en el proceso de generación de pruebas en la industria aeroespacial. Sirve como referencia para el análisis y discusión de los resultados obtenidos en el presente trabajo, así como para la formulación de recomendaciones y futuras líneas de investigación.

3 Capítulo 3. Marco Metodológico

Este capítulo describe la metodología adoptada para evaluar *LLMs* en pruebas aeroespaciales. Se articula un enfoque alternativo con un tipo de investigación aplicada. Se plantea realizar una evaluación de los modelos *LLMs* a partir de casos de estudio.

3.1 Tipo de Investigación

La investigación es aplicada y orientada a la solución de un problema práctico en un contexto regulado (DO-178C). Se asume la teoría de la complejidad y un pragmatismo metodológico que permite combinar mediciones objetivas y valoraciones expertas para obtener evidencia accionable.

3.2 Alcance Investigativo

El alcance es **descriptivo y exploratorio**. Además de lo señalado en los objetivos, se abordará explícitamente:

Descripción (estado actual y caracterización):

- **Proceso "as-is"** de identificación manual de casos de prueba en Avionyx (roles, artefactos, herramientas, tiempos por fase).
- **Tipología de requerimientos**: estructurados (p. ej., **EARS**) vs. no estructurados; atributos como ambigüedad, longitud y criticidad.
- **Métricas base**: tiempos actuales, precisión, exactitud, criterios de aceptación.

Exploración (oportunidades y limitaciones específicas):

- **Oportunidades**: reducción de tiempo en la identificación inicial de casos; mejora de cobertura por requerimiento; plantillas y prompts reutilizables; registro auditable de decisiones (prompts, versiones).

- **Limitaciones**: ambigüedad semántica, alucinaciones, reproducibilidad de resultados, cumplimiento DO-178C/DO-330 en el uso de herramientas, sesgos del modelo, seguridad y confidencialidad de datos, costos y latencia.

Resultados esperados del alcance: mapa de condiciones en que los *LLMs* aportan valor, umbrales mínimos de desempeño para su adopción parcial y lineamientos de integración con los procesos existentes.

3.3 Enfoque

El enfoque es alternativo ya que se enfoca en la evaluación de diferentes modelos y paradigmas de aplicación de LLMs por medio de casos de estudio. En este análisis, los resultados que se desean medir no pueden ser cubiertos únicamente con métricas cuantitativas ya que hay otros factores a tomar en cuenta tales como la calidad de las pruebas,evaluación de capacidades emergentes, coherencia,etc.

3.4 Diseño

El diseño de caso aplicado, flexible y adaptativo, con iteraciones tipo sprint. En cada iteración se comparan resultados, se ajustan prompts y flujos y se documentan decisiones.

3.5 Población y Muestreo

- **Población de referencia**: personal técnico de Avionyx vinculado a verificación de software en entornos DO-178C (ingenieros de verificación, de software y QA/Procesos).
- **Muestreo**: **no probabilístico por conveniencia**.

Cuantificación estimada *(ajustable según disponibilidad)*:

- **n ≈ 6–12 participantes**, distribuidos de la siguiente manera:

- **6–10** ingenieros/as de verificación.

- **Criterios de inclusión**: experiencia en DO-178C y participación en pruebas basadas en requerimientos.
- **Criterios de variación**: diversidad en años de experiencia, proyecto y rol.
- **Entrevistas cualitativas**: estimado 4–8 entrevistas.

3.6 Instrumentos de Recolección de Datos

Matrices de evaluación (con rúbricas y pesos)

Se aplicarán a cada requerimiento y al conjunto:

1. **Precisión** **(0–1)**
   Porcentaje de casos generados que verifican correctamente el requerimiento.
   - **1.0 - 0.80**:alta; **0.79 – 0.7**: media; **<0.7**: baja.
2. **Sensibilidad** **(0–1)**
   Porcentaje de casos generados que verifican correctamente el requerimiento con respecto a los casos faltantes no identificados por los *LLM*.
   - **1.0 - 0.80**:alta; **0.79 – 0.7**: media; **<0.7**: baja.
3. **Puntuación F1** **(0–1)**
   Balance entre precisión y sensibilidad. Combina ambas métricas en un solo número para obtener una representación general del desempeño.
   - **1.0 - 0.80**:alta; **0.79 – 0.7**: media; **<0.7**: baja.

(b) Entrevistas abiertas/semiestructuradas *(30–45 min)*

Ejes y ejemplos de preguntas:

- **Utilidad**: “¿En qué escenarios considera que los LLM son útiles para generar casos de prueba y en cuáles no? ¿Por qué?”
- **Carga cognitiva y flujo de trabajo**: “¿El LLM reduce su carga o introduce pasos extra?”
- **Riesgos/ética**: “¿Qué riesgos ve para seguridad/cumplimiento?”
- **Adopción**: “¿Qué umbrales y controles pondría para uso en producción?”

(c) Encuestas

Ítems Likert (1–5) sobre: utilidad percibida, facilidad de uso, claridad de casos, intención de uso.

(d) Registros de tiempo y eficiencia

- **Definición operacional**: tiempo desde recepción del requerimiento hasta tener lista la **lista de casos iniciales** (no ejecución).
- **Comparación**: manual vs. LLM por requerimiento.

3.6.1 3.7 Técnicas de Análisis de Información

- **Cuantitativo**: estadísticas descriptivas (precisión, sensibilidad), comparación pareada manual vs. LLM (ganancia de tiempo y diferencia en precisión/sensibilidad).
- **Cualitativo**: Clasificación de errores, relación entre métricas y tipos de requerimientos.

3.7 Técnicas de Análisis de Información

Las técnicas de análisis de información toman los datos recolectados en 3.6 y les dan un sentido útil para efectos de la investigación. Para este propósito, se pueden emplear técnicas muy diversas, como:

- Diagramas de flujo de datos
- Árboles de causa efecto
- Espinas de Ishikawa
- Diagramas BPMN y/o UML
- Mapas conceptuales
- Mapas mentales.

4 Capítulo 4. Análisis del Diagnóstico

El análisis de diagnóstico constituye una investigación previa sobre el tema de generación de casos de prueba en la industria aeroespacial con el fin de considerar aspectos importantes previo al desarrollo de este trabajo.

Dentro de los temas a considerar se encuentran los criterios de aceptación de los casos de prueba, normativas dentro de la industria aeroespacial, regulaciones del uso de inteligencia artificial e impacto en la calidad y seguridad de las pruebas.

La cualificación de herramientas es necesaria cuando las actividades de desarrollo o verificación de software se reducen, eliminan o automatizan mediante herramientas cuyo resultado no es verificado según lo descrito en la guía DO-178C (RTCA, 2011a).

Los errores introducidos por este tipo de herramientas pueden afectar negativamente la funcionalidad del software y la seguridad del sistema. El proceso de cualificación de herramientas proporciona guías para el desarrollo y la verificación de herramientas con el fin de asegurar que su integridad y funcionalidad prevista no se vean comprometidas (RTCA,2011b). El proceso y los objetivos que deben seguirse están documentados en la DO-330.

Durante este proceso, se debe demostrar que las salidas de las herramientas cumplen con los objetivos descritos en la DO-330. Las salidas deben ser consistentes cuando se proporcionan las mismas entradas en el mismo entorno. Para las salidas que muestran variaciones, se debe demostrar que no tienen un impacto negativo en el proceso que utiliza dichas salidas.

Dado que los LLM generan contenido basado en probabilidades, las herramientas impulsadas por LLM no deben considerarse para reemplazar, eliminar o automatizar ningún proceso de verificación de software, ya que no es posible obtener salidas determinísticas.

La FAA ha establecido una hoja de ruta que define los principios rectores para garantizar la seguridad de la inteligencia artificial en la aviación (Federal Aviation Administration, 2024). La hoja de ruta enfatiza la importancia de centrarse en la garantía de la seguridad y en las mejoras de seguridad, particularmente en áreas donde las aplicaciones de IA tienen el potencial de mejorar la seguridad operativa. También destaca la importancia de evitar la personificación de los sistemas de IA, ya que estas tecnologías no deben considerarse ni tratarse como seres humanos. Las decisiones críticas y el control total de sistemas o procesos no

deben delegarse a herramientas de IA. En su lugar, los diseñadores e ingenieros deben diferenciar claramente las responsabilidades asignadas a los sistemas de IA de aquellas asignadas a los operadores humanos para evitar cualquier compromiso con la seguridad.

La FAA recomienda adoptar un enfoque incremental para integrar la IA en los procesos de aviación. Esta estrategia progresiva permite a los especialistas y a la FAA acumular conocimiento y experiencia a partir de implementaciones de menor escala antes de desplegar sistemas basados en IA en dominios más críticos para la seguridad o de mayor impacto.

La guía DO-178C establece los criterios de aceptación que deben utilizarse como marco de referencia para la evaluación de resultados obtenidos por medio de *LLM.* El objetivo es analizar los resultados con mayor objetividad para determinar el valor aportado por herramientas que utilizan inteligencia artificial.

El análisis de diagnóstico revela la situación actual de las normativas en la industria aeroespacial con respecto a la verificación de software e implementación de herramientas impulsadas por IA en torno a esta actividad.

Se concluye que debido al impacto que tiene las actividades de verificación de software, la naturaleza probabilística de los *LLM* y las sugerencias por parte de la FAA, no se debe diseñar herramientas que busquen automatizar actividades de verificación.

El ser humano tiene la decisión final y es responsable por cada prueba desarrollada. Por lo tanto, este trabajo se enfoca en cómo herramientas impulsadas por IA pueden asistir al ser humano y agilizar los procesos de verificación sin comprometer la calidad de las pruebas.

También, se seleccionó un grupo de ingenieros al cual se le realizaron pruebas preliminares para analizar el tipo de ayuda que este tipo de herramientas debe brindar.

Los ingenieros se dividieron en tres categorías dependiendo de su nivel de experiencia en el área de verificación: menos de 1 año, de 1 año a 3 años y más de

3 años. El objetivo es identificar relaciones entre la experiencia y la calidad de casos de prueba identificados.

Se utilizaron las mismas métricas de precisión y sensibilidad descritas en este documento, así como el tiempo que se necesitó para identificar los casos de prueba.

Los resultados preliminares muestran que hay una relación entre el grado de experiencia de los ingenieros y la calidad de pruebas identificadas. Los ingenieros con más de tres años de experiencia obtuvieron una puntuación F1 promedio de 0.88. Los ingenieros entre uno y tres años de experiencia obtuvieron una puntuación F1 promedio de 0.8, mientras que los ingenieros con menos de un año de experiencia obtuvieron una calificación de 0.55.

5 Capítulo 5. Propuesta de Solución

Con base en la investigación realizada en este documento, se presenta a continuación, la propuesta para cumplir con los objetivos de este trabajo.

Se realizará una evaluación de tres paradigmas de aplicación de los LLMs para asistir a los ingenieros en la generación de casos de prueba. Los paradigmas de aplicación son: RAG, *prompts* y agentes.

Se evaluarán las respuestas obtenidas utilizando el checklist de la Tabla 2 y de acuerdo a este criterio de evaluación se emplearán las métricas de precisión, sensibilidad y puntaje F1 para analizar el desempeño de cada uno de los paradigmas y modelos empleados.

Los resultados obtenidos se compararán con las evaluaciones previas que se realizaron a los ingenieros con el fin de determinar el impacto que estas herramientas pueden tener en la productividad de los ingenieros. Adicionalmente, se realizarán entrevistas para analizar el impacto que tienen estas herramientas en el trabajo de los ingenieros.

Finalmente, se propone realizar un diseño de una aplicación que utilice *LLMs* para asistir a los ingenieros en la generación de casos de prueba. El diseño

de esta aplicación debe estar orientado a asistir al ingeniero y no reemplazarlo o automatizar estas tareas para cumplir con los estándares requeridos. El humano debe estar en flujo de trabajo de la aplicación para revisar y modificar los casos de prueba sugeridos por la aplicación.

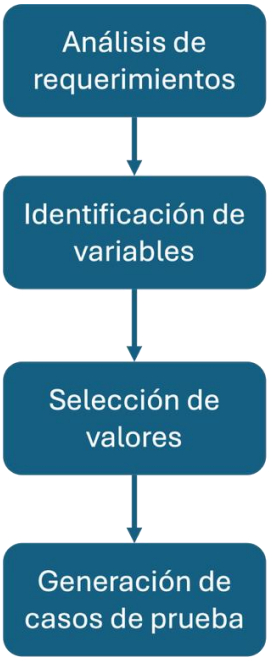
6 Capítulo 6. Proceso de identificación de casos de prueba

De acuerdo con la guía DO-178C, se espera que se incluya un conjunto mínimo de casos de prueba al seleccionar los casos de prueba normales y de robustez para un dado requerimiento. Este conjunto de casos de prueba depende de la funcionalidad, las variables y la lógica descritas en el requerimiento (RTCA, 2011).

A pesar de que los pasos a seguir cambian según el requerimiento, se propone el proceso mostrado en la para modelar el flujo de actividades necesarias que típicamente se siguen cuando se identifican casos de prueba.

En este flujo hay cuatro grandes actividades que modelan el proceso de identificación de casos de prueba. En las siguientes secciones, se describe cada actividad por separado.

Figura 1: Proceso de identificación de casos de prueba



6.1 Análisis de requerimientos

El proceso de verificación comienza cuando al ingeniero se le asigna uno o más requerimientos a los cuáles se les debe desarrollar las pruebas. Una vez que los requerimientos fueron asignados, el ingeniero estudia el sistema para obtener información que le permita identificar los casos de prueba necesarios y comprender la funcionalidad que se va a verificar. Dicha información puede encontrase en otros requerimientos, documentos que describen los protocolos de comunicación utilizados, documentos que describe las interfaces de software y hardware, etc.

Este proceso de búsqueda de información corresponde al proceso de análisis de requerimientos mostrado en la Figura 1. En este proceso se obtiene información relevante como por ejemplo el rango de operación de las señales del sistema, tolerancias, conflictos y dependencias con otros requerimientos, modos de operación del sistema, etc.

Debido a que este primer paso implica el análisis de múltiples documentos y la identificación de la relación entre los requerimientos a verificar y el contenido de dichos documentos, se considera que esta actividad se encuentra fuera del alcance de la presente evaluación. Incluir este paso añadiría un nivel adicional de

complejidad que podría introducir errores o sesgos en la actividad principal objeto de estudio: la identificación de casos de prueba.

En los requerimientos utilizados en esta investigación se incluye la información necesaria para identificar los casos de prueba de manera directa. Por esta razón, el proceso de analizar requerimientos, agruparlos según la funcionalidad que describen y complementar la información mediante otros documentos se propone como una línea de investigación para trabajos futuros, y no se contempla dentro del enfoque del presente estudio.

6.2 Identificación de variables

Las tres variables esenciales de un requerimiento son las condiciones o variables de entrada, los resultados esperados y las condiciones iniciales (cuando aplique).

Las condiciones de entrada representan las variables definidas en un requerimiento que determinan si deben ocurrir acciones o comportamientos específicos del sistema. Por ejemplo, en un requerimiento que establece: “Si el sistema está en el Aire y la altitud supera los 10,000 ft, entonces el sistema deberá…”, las condiciones de entrada corresponden al estado del sistema y a la altitud. Si estas condiciones no se cumplen, el sistema no ejecuta la acción especificada.

Los resultados esperados representan las variables o respuestas del sistema definidas en un requerimiento que indican el resultado esperado de una acción o comportamiento específico. Por ejemplo, en un requisito que establece: “Si el sistema está en el aire y la altitud supera los 10,000 ft, entonces el sistema deberá ajustar la presión de cabina a 11 psi”, la salida corresponde a la activación del mecanismo de control de presión que establece la presión de cabina. Estas variables de salida sirven como indicadores observables del comportamiento del sistema bajo las condiciones de entrada especificadas.

Las condiciones iniciales se refieren a estados o condiciones que deben configurarse antes de ejecutar el caso de prueba, y tienen varios propósitos. Primero, esto previene falsos positivos. Por ejemplo, si los resultados previos de

casos de prueba no se eliminan, las observaciones en los siguientes casos podrían verse afectadas por condiciones anteriores. Además, garantiza que las condiciones de observación estén establecidas antes de introducir las entradas en el sistema bajo prueba.

Para visualizar todos los escenarios que deben considerarse en el procedimiento, esta información se presenta comúnmente en una tabla de verdad. Las columnas incluyen la siguiente información: ID del caso de prueba, la descripción del propósito, el tipo de prueba, las condiciones iniciales y los valores de entrada y salida.

6.3 Selección de valores

Una vez que se extraen las variables de entradas, el proceso continúa seleccionando los valores que deben probarse para cada una.

DO-178C define una guía con respecto a los aspectos que deben considerarse al seleccionar casos de prueba normales y de robustez para un requerimiento determinado. Estas consideraciones se incorporan en los planes de verificación del proyecto para asegurar que el proceso de revisión de pruebas las contemple adecuadamente. En este estudio, la lista de verificación incluida en la Tabla 2 se considera como la lista que indica si las pruebas son correctas y si existen casos de prueba faltantes.

En la Tabla 2 se puede observar que existen diferentes elementos de la lista de verificación aplicables a distintos tipos de entradas. Por ejemplo, si la entrada es numérica, entonces deben considerarse los límites, así como los valores máximos y mínimos.

En esta investigación se proponen tres categorías de entradas: booleana, rango continuo y enumeración. Las entradas booleanas corresponden a variables de entrada que solo tienen dos valores posibles: encendido/apagado, verdadero/falso, presente/no presente, inválido/válido, etc. Las entradas de rango continuo son variables que pueden asumir cualquier valor dentro de un intervalo definido, como temperatura, voltaje, altitud o velocidad. Las entradas de enumeración son variables que solo pueden asumir un conjunto predefinido de

valores discretos, representando típicamente modos del sistema, estados u opciones categóricas. Algunos ejemplos incluyen configuraciones, modos operacionales o indicadores de estado que están limitados a valores específicos enumerados.

La Tabla 3 muestra los aspectos que deben considerarse para cada tipo de entrada. Los LLMs deben ser instruidos para seleccionar valores de prueba que satisfagan dichas consideraciones, con el fin de aprobar la revisión de los casos de prueba.

Tabla 3: Valores esperados según el tipo de entrada

Tipo de variable de entrada	**Valores a considerar**
Booleana	Las entradas booleanas solo tienen dos valores posibles. Ambos deben considerarse en los casos de prueba finales para cubrir cualquier clase de equivalencia asociada con estas entradas y el rango de la entrada.
Rango continuo	Los valores seleccionados para este tipo de entrada deben cubrir las clases de equivalencia, los rangos de entrada y las condiciones de frontera. Estas entradas también deben considerar la resolución de la variable y la tolerancia especificada en el requerimiento, con el fin de calcular correctamente los valores de entrada para los límites. En caso de que existan operaciones matemáticas involucradas, los valores de entrada también deben considerar escenarios de robustez para valores que podrían generar resultados indeterminados, tales como divisiones entre cero, posibles condiciones de desbordamiento/subdesbordamiento, y valores especiales que pueden representarse mediante variables de precisión simple y de doble precisión.
Enumeración	Los valores seleccionados para este

Tipo de variable de entrada	Valores a considerar
	tipo de entrada deben contener todos los valores enumerados definidos en el requerimiento, además de una cláusula de “otros” (cuando aplique).

6.4 Generación de casos de prueba

Los valores de entrada que fueron seleccionados se combinan para generar los casos de prueba. Cada caso de prueba contiene los valores de entrada que se evalúan junto con el resultado esperado correspondiente.

El paso final de este proceso consiste en tomar los casos de prueba que fueron generados, junto con la información correspondiente de las entradas, los resultados esperados y las condiciones iniciales, para construir una tabla de verdad que resuma los casos de prueba generados para un requerimiento. Además de las entradas, los resultados esperados y las condiciones, se agregan dos columnas adicionales para indicar el ID del caso de prueba y la descripción de la prueba.

7 Capítulo 7. Clasificación de requerimientos

Existen diversas categorías de representación de requerimientos para la generación de pruebas basadas en requerimientos: lenguaje natural, semi-formal, basado en modelos, formal matemático e híbrido. Este estudio se enfoca en representaciones en lenguaje natural, donde los casos de prueba son escenarios documentados en un formato tabular, por ejemplo, tablas de verdad o tablas de decisión (Yang, Huang, Cui, Niu, & Towey, 2025).

Las descripciones de los requerimientos pueden incluir operadores booleanos, múltiples condiciones, máquinas de estados, algoritmos, funcionalidades temporales, etc. Debido a la amplia variabilidad y a las posibles combinaciones de este tipo de requerimientos, desarrollar una única solución capaz de procesar todas las formas posibles representa un desafío significativo.

La Tabla 4 muestra las clasificaciones que se proponen para procesar los diferentes tipos de requerimientos que pueden encontrarse. Cada tipo de requerimiento representa una estrategia única que se seguirá al desarrollar los casos de prueba.

Tabla 4: Tipos de requerimientos

Tipo	Descripción
Con condiciones	El requerimiento describe las condiciones bajo las cuales el software debe implementar una determinada acción. Cuando se indican múltiples condiciones, estas se unen mediante operadores booleanos o equivalentes para describir completamente la lógica del requerimiento.
Sin condiciones	El requerimiento describe acciones que el software debe implementar incondicionalmente.
Máquina de estados	El requerimiento describe una máquina de estados, típicamente mediante diagramas e imágenes. En estas representaciones se explica el flujo de trabajo del proceso, incluyendo todas las condiciones necesarias para cambiar de un estado a otro. Además, se requieren varios requerimientos para verificar completamente la máquina de estados. De acuerdo con DO-178C, los casos de prueba para este tipo de requerimiento deben verificar que se ejerciten todas las transiciones posibles y que no se realicen transiciones de estado inválidas.

La idea de cada categoría no es únicamente seleccionar las estrategias de prueba, sino también indicar qué elementos de la lista de verificación de la deben considerarse para asegurar que los casos de prueba sugeridos cumplan con las directrices de DO-178C.

Dado que los requerimientos de máquina de estado involucran procesar varios requerimientos simultáneos o incluso procesar imágenes, se consideran objeto de estudio para futuras investigaciones. En esta investigación se desea comprender las limitaciones, desafíos y oportunidades de utilizar LLMs para identificar casos de prueba por lo que agregar complejidades adicionales pueden alterar los resultados. Además, se sigue el principio de trabajo iterativo e

incremental propuesto en la hoja de ruta del uso de IA en la industria aeroespacial establecida por la FAA (Administration, 2024) .

El proceso detallado en el Capítulo 6. Proceso de identificación de casos de prueba aplica para requerimientos representados en lenguaje natural cuyo tipo sea con condiciones o sin condiciones.

8 Capítulo 8. Caso de estudio del proceso de identificación de casos de prueba

El proceso seguido para generar casos de prueba en este experimento se explica en esta sección mediante un ejemplo, incluyendo los detalles de cada paso y el formato esperado de la salida.

Dado el siguiente requerimiento:

“Durante la operación normal, si el interruptor hidráulico está encendido y la presión está por debajo de 2500 psi, el sistema deberá calcular el ciclo de trabajo de la bomba utilizando Ciclo de Trabajo = (2500 – Presión) / 500 × 10%; de lo contrario, deberá establecer el Ciclo de Trabajo en 0. El rango de presión va de 0 a 5000 psi con una resolución de 0.1 psi y una tolerancia de +/- 1 psi. La tolerancia para el ciclo de trabajo PWM es de +/- 1%.”

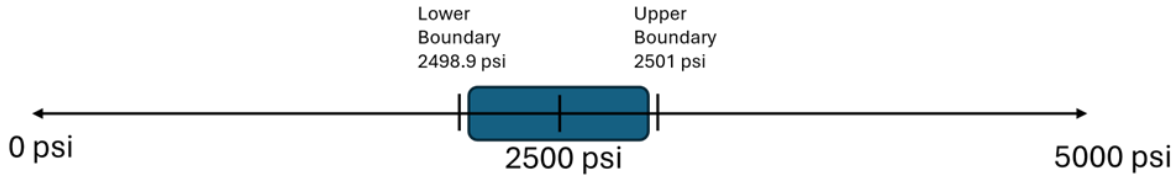
El primer paso es clasificar el requerimiento dentro de una de las tres categorías. Dado que este requerimiento involucra condiciones, debe clasificarse como “con condiciones”.

Las variables del requerimiento para este escenario son: operación normal (condiciones iniciales), interruptor hidráulico (entrada), presión (entrada) y ciclo de trabajo de la bomba (salida).

El interruptor hidráulico es una entrada booleana con dos valores posibles: encendido y apagado. La presión es una entrada de tipo rango continuo que debe considerar varios aspectos, como el rango de entrada y los límites descritos en el requerimiento.

La Figura 2 muestra el rango de entrada para la presión, donde pueden observarse los valores mínimo y máximo del rango, así como el umbral en 2500 psi. Dado que existe una tolerancia de +/- 1 psi, se recomienda no utilizar valores dentro del rango resaltado en la Figura 2; de lo contrario, los casos de prueba podrían producir resultados diferentes en distintas iteraciones de la ejecución del procedimiento de prueba. Para garantizar que los valores que verifican los límites inferior y superior siempre estén por debajo y por encima del umbral, respectivamente, deben considerarse la tolerancia y la resolución.

Figura 2: Rango de tolerancia



El proceso general consiste en considerar el límite inferior como (límite – tolerancia – resolución) y el límite superior como (límite + tolerancia + resolución), aunque la redacción del requerimiento puede especificar si la resolución debe aplicarse. En este caso, los valores de prueba de frontera son 2498.9 y 2501.1. Sin embargo, debido a que el requerimiento establece “por debajo de 2500 psi”, un valor de prueba de 2501 también es aceptable: incluso con una tolerancia de ±1 psi, siempre se evaluará como un valor mayor o igual a 2500 psi.

Para seleccionar la combinación de valores de entrada para los casos de prueba, se evalúa una entrada a la vez mientras se establece un valor fijo para las demás entradas. Esto garantiza que la entrada bajo prueba influya en el valor de salida. En este caso, se iteran todos los valores de entrada de presión mientras se establece el interruptor hidráulico en encendido. Una vez que se seleccionan todos los valores de presión, se elige un valor menor a 2500 psi mientras se iteran todos los valores posibles del interruptor hidráulico. Los casos de prueba duplicados se descartan.

Finalmente, los casos de prueba se presentan en un formato tabular, como se muestra en la **Error! Reference source not found.**, donde se especifican el ID

del caso de prueba, la descripción del propósito, el tipo de prueba, las condiciones iniciales y los valores de entrada y salida.

Tabla 5: Tabla de casos de prueba

ID del caso de prueba	Descripción	Tipo	Condiciones iniciales	Variables de entrada	Variables de salida
TC-001	Prueba el valor mínimo de la Presión.	Normal	Operación normal	Presión = 0, Interruptor hidráulico = Encendido	Ciclo de trabajo = 50 +/- 1%
TC-002	Prueba el valor máximo Presión.	Normal	Operación normal	Presión = 5000, Interruptor hidráulico = Encendido	Ciclo de trabajo = 0 +/- 1%
TC-003	Prueba el límite inferior de la Presión.	Normal	Operación normal	Presión = 2498.9, Interruptor hidráulico = Encendido	Ciclo de trabajo = 0 +/- 1%
TC-004	Prueba el límite superior de la Presión.	Normal	Operación normal	Presión = 2501 Interruptor hidráulico = Encendido	Ciclo de trabajo = 0 +/- 1%
TC-005	Prueba un valor medio de la Presión.	Normal	Operación normal	Presión = 1000, Interruptor hidráulico =	Ciclo de trabajo = 30 +/- 1%

ID del caso de prueba	Descripción	Tipo	Condiciones iniciales	Variables de entrada	Variables de salida
				Encendido	
TC-006	Prueba un valor medio de la Presión.	Normal	Operación normal	Presión = 3000, Interruptor hidráulico = Encendido	Ciclo de trabajo = 0 +/- 1%
TC-007	Prueba el interruptor cuando está apagado	Normal	Operación normal	Presión = 2498.9, Interruptor hidráulico = Apagado	Ciclo de trabajo = 0 +/- 1%

9 Capítulo 9. Descripción del experimento

Para llevar a cabo esta evaluación, es necesario definir un *benchmark* de requerimientos con sus correspondientes casos de prueba esperados. Se crearon cerca de 90 requerimientos que describen funcionalidades utilizando lenguaje natural e incluyen variables booleanas, enumeradas y de rango continuo. Los requerimientos siguen la sintaxis EARS para garantizar requerimientos de alta calidad en donde se sigue una estructura definida para identificar las condiciones y acciones esperadas (Mavin, 2019).

Los requerimientos del *benchmark* se distribuyeron en dos categorías: validación y pruebas. Los requerimientos de validación se utilizaron para probar los *prompts* y código utilizado en fases iniciales del experimento. Los requerimientos de pruebas se utilizaron para los experimentos que se llevaron a cabo para recolectar los resultados del estudio.

Para cada requerimiento dentro del grupo de pruebas, se identificó los casos mínimos de pruebas que se esperan. Posteriormente, se utilizó el checklist propuesto en la Tabla 2 para verificar las respuestas. Tres métricas cuantitativas

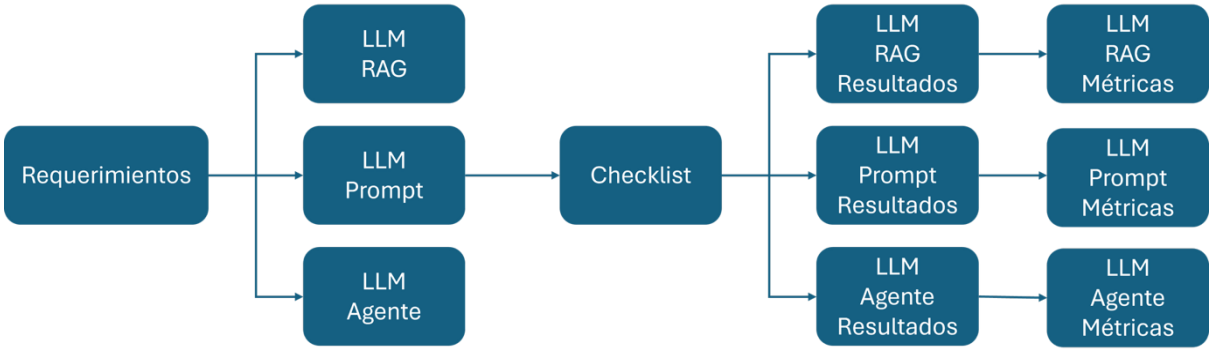
(precisión, puntuación F1 y sensibilidad) se utilizan para analizar los casos de prueba.

Para los casos de prueba faltantes o incorrectos, se documentan los tipos de errores identificados con el fin de proporcionar una evaluación cualitativa de los casos de prueba generados.

En este estudio, se realizaron dos evaluaciones para analizar las respuestas obtenidas por los diferentes paradigmas de aplicación.

La Figura 3 muestra el proceso seguido en la primera evaluación. Primero se definen los requerimientos de prueba que se van a utilizar. Los diferentes paradigmas de aplicación generan los casos de prueba para dichos requerimientos. Una vez que se obtienen los resultados, se sigue el checklist propuesto en la Tabla 2 y se evalúan los resultados manualmente. Durante la evaluación de los resultados se identifican los casos de prueba correctos, incorrectos y faltantes; además, de los tipos de errores encontrados. Como último paso, se generan las métricas para cada paradigma.

Figura 3: Experimento LLM vs LLM



El segundo experimento es una evaluación de tres preguntas en donde, se le solicita a un grupo de ingenieros, identificar los casos de prueba para cada pregunta. Los ingenieros que son parte de la evaluación de clasifican en tres categorías dependiendo de su nivel de experiencia: menos de 1 año, entre 1 año y 3 años y más de 3 años.

Los requerimientos utilizados en las preguntas también se emplean para generar casos de prueba con los diferentes paradigmas de aplicación. El propósito de esta evaluación es comparar el desempeño de los paradigmas de aplicación con el de ingenieros para identificar limitaciones, ventajas de utilizar LLMs y oportunidades de mejora.

Diferentes LLMs varían en sus arquitecturas, datos de entrenamiento y estrategias de optimización, lo cual puede influir significativamente en su desempeño a través de distintas tareas. Además, dado que es común que los requerimientos en los proyectos se clasifiquen como datos propietarios, las organizaciones a menudo prefieren utilizar LLMs de código abierto internamente en lugar de depender únicamente de servicios externos. Por lo tanto, este estudio adopta un enfoque híbrido, investigando tanto un modelo de código abierto como un modelo comercial para evaluar su desempeño comparativo.

Los modelos seleccionados para la evaluación son GPT-5 y Mistral. GPT-5 es un modelo comercial disponible en la empresa donde se realiza este estudio. Mistral fue seleccionado con base en su uso previo en estudios relacionados, en los cuales se demostraron su desempeño y resultados de interés (Yang et al., 2025; Alassan et al., 2024) .

Un desafío importante de los LLMs es que pueden generar información incorrecta. Este fenómeno se denomina alucinación y ocurre porque los LLMs predicen palabras basándose en patrones aprendidos durante el entrenamiento, sin considerar ninguna verdad fundamental. Si el usuario realiza una pregunta que el modelo nunca ha visto antes, este hará una suposición estadística basada en los patrones observados durante su entrenamiento. Este proceso puede llevar a la producción de información inexacta que parece coherente y convincente, a pesar de ser incorrecta (Koon, 2024).

Para considerar este fenómeno en la evaluación, el mismo requerimiento será evaluado tres veces para cada modelo y estrategia de prompt, con el fin de analizar la consistencia, la posible presencia de alucinaciones y variaciones en las respuestas.

En resumen, para considerar las variables adicionales previamente mencionadas, se debe obtener los resultados de cada requerimiento para cada combinación. Por ejemplo, para el requerimiento 1 se obtiene los resultados para las siguientes variaciones:

- RAG con GPT-5. Se obtienen casos de prueba para el requerimiento 1 tres veces.
- RAG con Mistral. Se obtienen casos de prueba para el requerimiento 1 tres veces.
- Prompt con GPT-5. Se obtienen casos de prueba para el requerimiento 1 tres veces.
- Prompt con Mistral. Se obtienen casos de prueba para el requerimiento 1 tres veces.
- Agente con GPT-5. Se obtienen casos de prueba para el requerimiento 1 tres veces.
- Agente con Mistral. Se obtienen casos de prueba para el requerimiento 1 tres veces.

10 Capítulo 10. Análisis de resultados

Como se explicó en el Capítulo 9. Descripción del experimento, durante la primera parte del experimento se evalúan los casos de prueba generados por las diferentes combinaciones de LLMs y los paradigmas de aplicación.

Posteriormente, se realiza una segunda evaluación en donde se comparan los casos de prueba identificados por ingenieros y los diferentes paradigmas de aplicación de los LLMs.

A continuación, se analiza los resultados obtenidos en cada parte del experimento.

10.1 Experimento LLMs

La primera parte de la evaluación involucra 2 modelos diferentes con tres paradigmas de aplicación distintos.

Durante este análisis, los casos de prueba sugeridos por las diferentes combinaciones de LLMs y paradigmas se comparan para analizar las variaciones entre las métricas de precisión y sensibilidad.

Los requerimientos utilizados para esta evaluación son ficticios y fueron creados considerando las diferentes variaciones en la redacción para especificar condiciones, límites y la lógica para relacionar las entradas con las salidas.

Los identificadores de los requerimientos indican el tipo de lógica involucrada con el fin de analizar posteriormente los resultados obtenidos. Dentro de las categorías que se consideraron se encuentran:

- ALG: operadores matemáticos
- MIX: múltiples condiciones
- REL: condiciones relacionales
- BOOL: entradas booleanas
- NOCND: sin condiciones

10.1.1 RAG

La motivación de elegir este paradigma de aplicación en esta evaluación radica en estudiar posibles implementaciones por parte de empresas que buscan proteger sus requerimientos y documentación. En ocasiones los requerimientos y otros datos de las empresas tales como buenas prácticas, documentación, lecciones aprendidas no puede compartirse públicamente. RAG ofrece una alternativa a estos casos ya que se les proporciona a los modelos información adicional bajo la cual no fueron entrenados.

La Figura 4 presenta el flujo de la aplicación basada en el paradigma RAG utilizada en esta evaluación. En este proceso, el módulo correspondiente a la base de datos vectorial desempeña un papel fundamental en el procesamiento de documentos externos que se incorporan como contexto para la generación de casos de prueba. Esta base de datos se construye a partir de la transformación del contenido textual en representaciones vectoriales denominadas *embeddings*.

Dichos *embeddings* constituyen una representación numérica densa del texto, en la cual cada dimensión del vector codifica características latentes del contenido, tales como información semántica, relaciones sintácticas, patrones de significado, etc (Barnard, 2026). Esto permite descomponer el texto en números que posteriormente se pueden utilizar para analizar similitudes de palabras y oraciones. Bajo este principio es que las aplicaciones RAG responden preguntes y buscan información relevante de acuerdo a las instrucciones del usuario.

Estos embeddings se generan a partir de un modelo de embeddings. Para el caso de la combinación que utiliza GPT-5 se empleó el modelo de Azure *text-embeddings-3-large*, mientras que para la combinación que utiliza Mistral se usó el modelo de Hugging Face *sentence-transformers/all-MiniLM-L6-v2.*

Los *embeddings* generados a partir de los documentos se almacenan en segmentos previamente definidos por la aplicación. El tamaño de dichos segmentos es configurable, lo que permite ajustar la cantidad de palabras contenidas en cada uno. Esta configuración es relevante, ya que un segmento excesivamente grande puede integrar información perteneciente a párrafos con ideas no necesariamente relacionadas, afectando la precisión semántica. Por el contrario, un segmento demasiado pequeño puede fragmentar una misma idea y omitir información contextual importante.

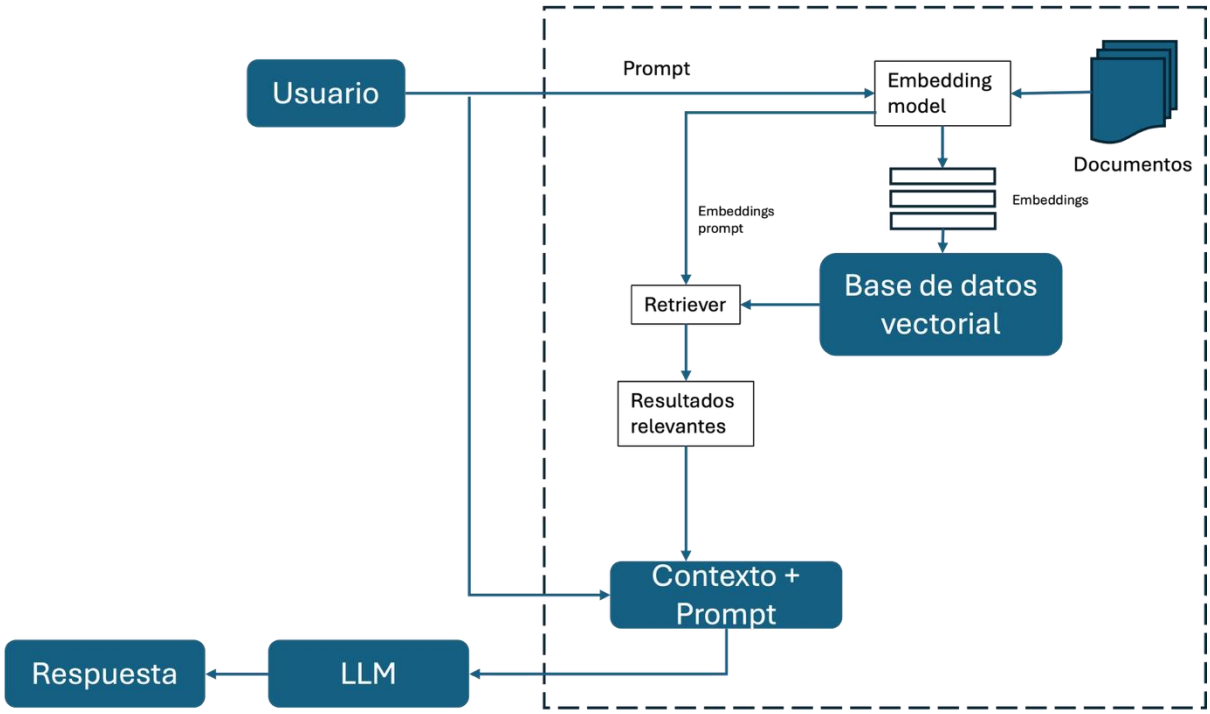
Los segmentos de embeddings se almacenan en la base de datos vectoriales de manera local utilizando *Chroma* (Chroma, 2026).

El proceso inicia con la incorporación de uno o varios documentos que se desean utilizar como fuente de contexto adicional para el LLM. Estos documentos son procesados y almacenados en la base de datos vectorial mediante su conversión a *embeddings*. Una vez completada esta etapa, el sistema queda preparado para la generación de casos de prueba.

Cuando el usuario solicita la creación de casos de prueba a través de un *prompt*, este también es transformado en un *embedding*. Posteriormente, el módulo de recuperación (*retriever*) realiza una búsqueda semántica en la base de datos vectorial para identificar los *embeddings* más similares al del *prompt*.

Finalmente, los segmentos de texto asociados a los *embeddings* recuperados se utilizan como contexto adicional que se proporciona al LLM, permitiéndole generar respuestas con base a los documentos proporcionados.

Figura 4: Flujo de la aplicación RAG



En este caso, la información descrita en el Capítulo 6. Proceso de identificación de casos de prueba se incluyó en un documento adicional para que el LLM pudiera tener acceso a este proceso específico.

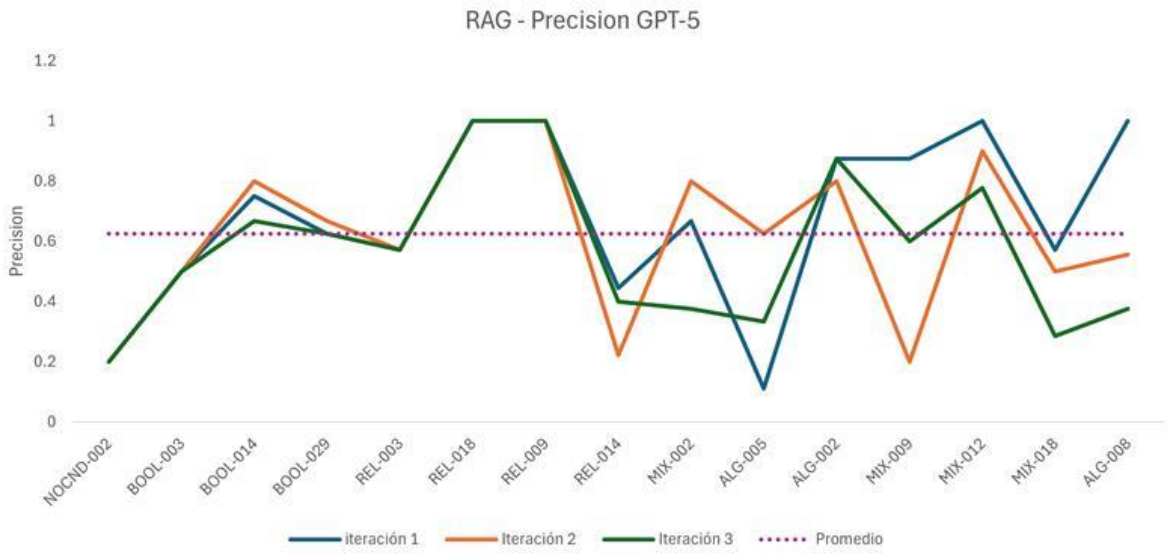
La Tabla 6 muestra los resultados obtenidos cuando el LLM se utilizó como una aplicación RAG. Cada requerimiento fue probado tres veces para evaluar la consistencia y estabilidad de los resultados generados.

Tabla 6: Resultados RAG

Combinación	**Precisión (promedio)**	**Sensibilidad (promedio)**	**Puntaje F1**
LLM: GPT-5	0.625	0.765	0.687

Modelo embedding **text-embedding-3-large (Azure OpenAI)**			
LLM: Mistral **Modelo embedding:** **all-MiniLM-L6-v2** **(Hugging Face)**	0.538	0.541	0.540

Figura 5: Precisión obtenida con RAG utilizando GPT-5



Al utilizar GPT-5, se obtuvieron métricas más altas de precisión y sensibilidad. La Figura 5 muestra la precisión de esta combinación para el conjunto

de requerimientos utilizados en la evaluación. Los requerimientos se organizaron en orden creciente de complejidad a lo largo del eje x. Los resultados indican que la consistencia de las salidas tiende a disminuir, mostrando una mayor variabilidad a medida que aumenta la complejidad de los requerimientos.

Mistral produjo, en general, menos casos de prueba correctos. Como se muestra en la Figura 7, su mejor desempeño ocurrió cuando los requerimientos involucraban únicamente entradas booleanas. Sin embargo, cuando los requisitos incluían variables con tolerancias, condiciones de frontera u operaciones matemáticas, la precisión disminuyó. También se puede observar que la diferencia entre iteraciones aumentó para este tipo de requisitos. El modelo a menudo no aplicó correctamente las tolerancias, no identificó de manera confiable los casos de prueba relacionados con límites y generó resultados incorrectos en operaciones matemáticas

Figura 6: Sensibilidad obtenida con RAG utilizando GPT-5

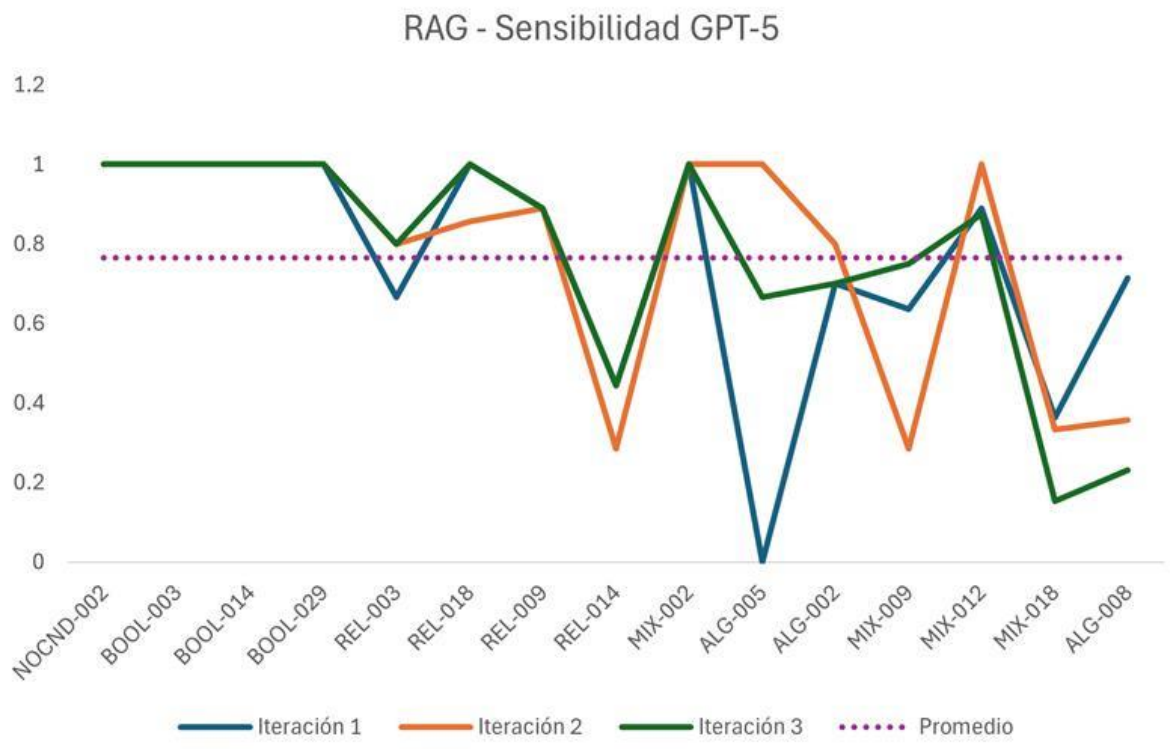
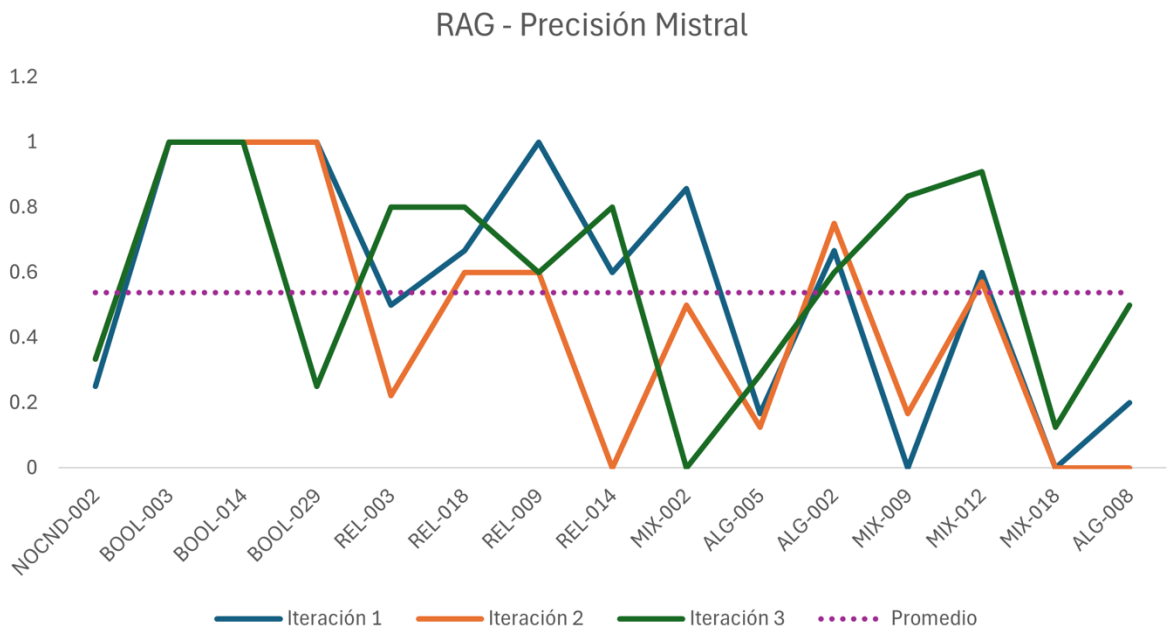


Figura 7: Precisión obtenida con RAG utilizando Mistral



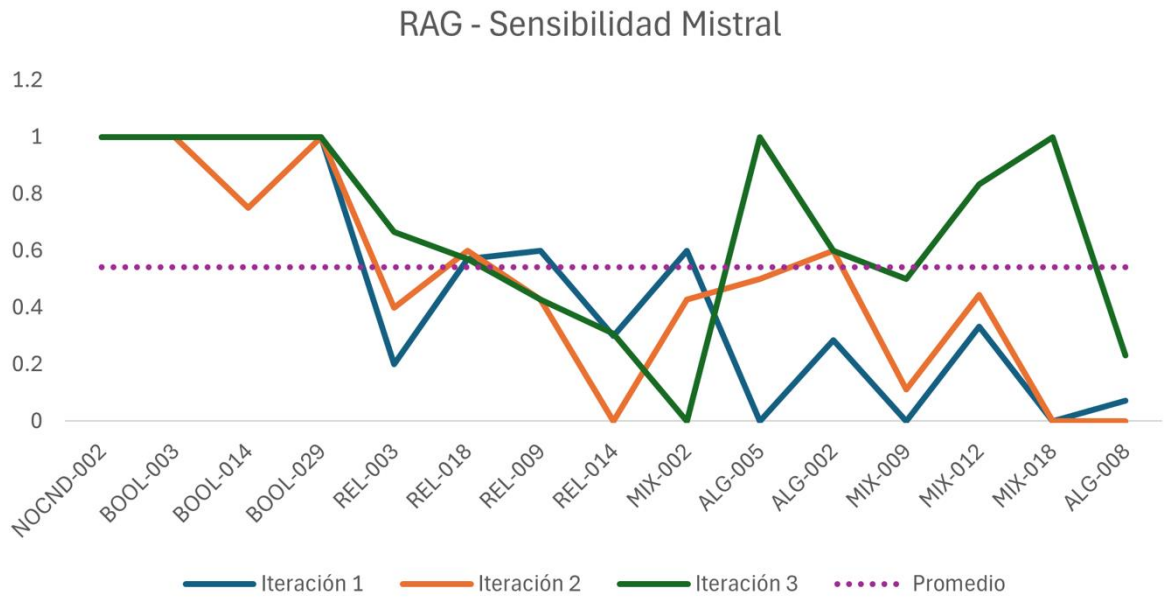
En la Figura 6 y la Figura 8 se pueden observar los resultados de sensibilidad para las tres iteraciones utilizando una aplicación RAG con GPT-5 y Mistral, respectivamente. Esta métrica representa los casos de prueba faltantes que el LLM no sugirió y que eran necesarios para cubrir el requerimiento y cumplir con la lista de verificación de la Tabla 2.

En ambos casos, la sensibilidad disminuye a medida que aumenta la complejidad de los requerimientos, lo que indica que se identificó un mayor número de casos de prueba faltantes en estos escenarios. La principal diferencia entre GPT-5 y Mistral radica en su cobertura: la combinación con GPT-5 fue capaz de sugerir la mayoría de los casos de prueba para un conjunto más amplio de requerimientos. Los valores más bajos de sensibilidad para GPT-5 se observaron en niveles más altos de complejidad de los requerimientos, desplazándose más hacia la derecha en comparación con la combinación con Mistral.

La combinación con Mistral omitió la mayoría de los casos de prueba cuando los requerimientos involucraban límites, pruebas de rango de entrada y pruebas de múltiples entradas. Como se puede observar en la Figura 8, cuando los requerimientos involucraban únicamente entradas booleanas, la sensibilidad fue en general cercano a 1; sin embargo, cuando se involucraron diferentes tipos de

entradas, disminuyó, lo que demuestra que esta combinación resultó útil principalmente en escenarios donde solo se consideran requerimientos con entradas booleanas.

Figura 8: Sensibilidad obtenida con RAG utilizando Mistral



La Tabla 7 muestra el número de casos de prueba que no fueron identificados por cada combinación y el número de casos de prueba que contenían errores. La cantidad de casos de prueba incorrectos fue similar entre ambas combinaciones. Sin embargo, la combinación con GPT-5 alcanzó una mayor precisión porque produjo un mayor número de casos de prueba correctos en comparación con la combinación con Mistral.

La principal diferencia entre ambas combinaciones radica en el número de casos de prueba omitidos, donde la combinación con Mistral superó a GPT-5 en un 78 %.

Tabla 7: Distribución de casos de prueba faltantes o incorrectos

Type of issue	RAG GPT-5	RAG Mistral
Missing test cases	94	168

Type of issue	RAG GPT-5	RAG Mistral
Incorrect test cases	124	112

Los casos de prueba incorrectos y faltantes que no fueron sugeridos por las aplicaciones RAG se desglosaron adicionalmente para comprender los desafíos y limitaciones de este paradigma de aplicación. Se propusieron las siguientes categorías para los casos de prueba faltantes e incorrectos:

• **Entradas inválidas**: Casos de prueba donde los valores de entrada sugeridos no estaban especificados en la descripción del requerimiento, estaban fuera del rango máximo indicado o incluían valores incorrectos que no formaban parte del tipo y rango de entrada.

• **Problemas en los límites**: Casos de prueba donde la tolerancia no se aplicó correctamente para evaluar los límites del requerimiento, o donde no se sugirieron pruebas para verificar umbrales y condiciones de frontera.

• **Rango completo no evaluado**: Las pruebas no consideraron los valores máximos, mínimos o intermedios dentro del rango especificado.

• **Resultados esperados incorrectos**: Los casos de prueba no predijeron correctamente el resultado esperado.

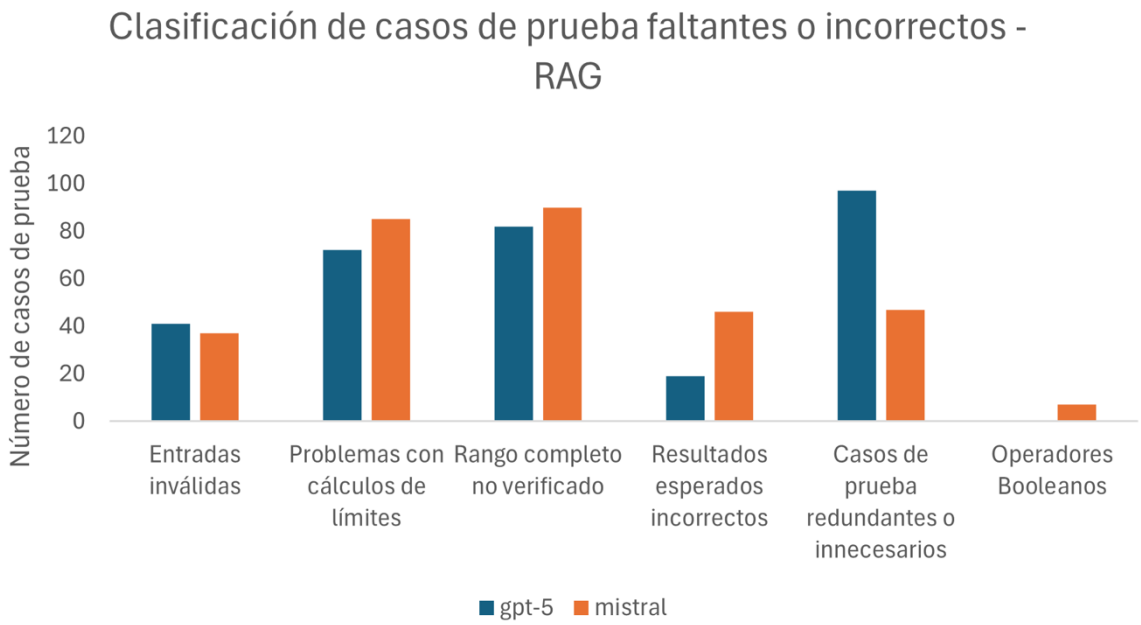
• **Operadores booleanos**: El caso de prueba no evaluó correctamente los operadores booleanos que unen las entradas en los requerimientos.

• **Casos de prueba repetidos o innecesarios**: Pruebas que están repetidas o que no aportan valor adicional. Estos casos podrían eliminarse sin afectar la cobertura del requerimiento. No se consideran incorrectos siempre que sus entradas y resultados esperados sean válidos; sin embargo, se monitorean porque generan esfuerzo innecesario para los ingenieros, quienes eventualmente deben evaluarlos y posiblemente eliminarlos.

La Figura 9 muestra que, para ambas combinaciones, los problemas relacionados con los límites y el rango completo no evaluado fueron los más comunes en los casos de prueba sugeridos por las aplicaciones RAG. Los requerimientos relacionados con límites deben incorporar la tolerancia, resolución y umbral especificados para identificar correctamente los casos de prueba necesarios que cubran condiciones de frontera y relacionales (por ejemplo, <, >). En estos escenarios, este paradigma de aplicación no resultó particularmente útil, ya que el proceso resumido por el recuperador RAG no incluía toda la información necesaria para calcular con precisión los valores límite. Además, los casos de prueba sugeridos no cubrieron el rango completo de entradas involucradas en el requerimiento, lo que implica que los ingenieros de verificación aún tendrían que agregar los casos de prueba faltantes no identificados por los LLMs.

En el caso de GPT-5, otro problema común observado fue el número adicional de casos de prueba sugeridos por el LLM para cubrir el requerimiento. Aunque los casos de prueba eran correctos, el sobretesteo de un requerimiento representa un esfuerzo adicional que puede evitarse. Los casos de prueba pasarán a formar parte del procedimiento de prueba, por lo que tener más casos de prueba de los necesarios incrementa el esfuerzo requerido para las actividades de desarrollo, revisión y mantenimiento.

Figura 9: Resumen de errores encontrados utilizando RAG



10.1.2 Prompt

Los *prompts* son las instrucciones utilizadas para indicar a los LLM que realicen las tareas solicitadas por un usuario. Existen buenas prácticas y técnicas de ingeniería de *prompts* que pueden utilizarse para mejorar la calidad de las respuestas generadas por los LLM.

En este caso, el *prompt* para generar casos de prueba está diseñado con base en las técnicas de ingeniería de *prompts* denominadas “Chain of Thought” (CoT) y “Role based” (Geromeinko, 2024). CoT es una técnica de *prompting* que incentiva al LLM a descomponer tareas complejas en pasos más pequeños con el fin de obtener mejores resultados. Esta técnica se alinea bien con la tarea de desarrollar casos de prueba, ya que requiere seguir una secuencia estructurada de pasos para lograr el objetivo general de la generación de casos de prueba. En el *prompt* también se indica que el LLM debe actuar como un ingeniero experto en verificación, con experiencia en las directrices DO-178C, con el fin de adaptar las respuestas a este tipo de actividad.

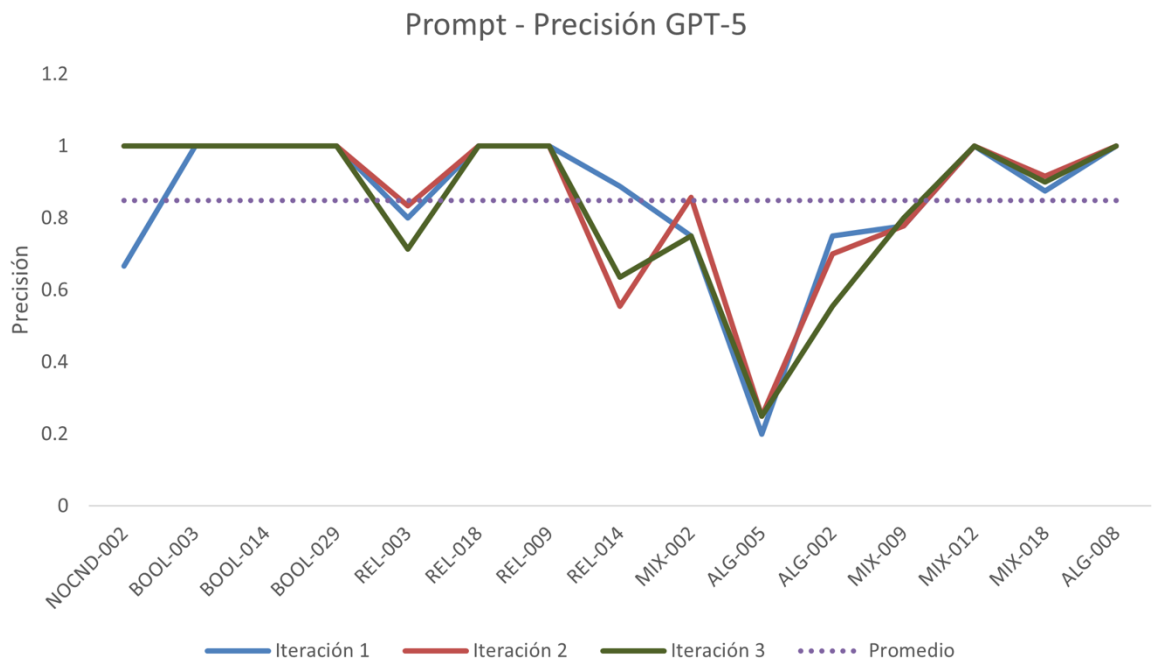
La Tabla 8 muestra las métricas obtenidas para el paradigma de aplicación de *prompts* al utilizar GPT-5 y Mistral. En el caso de GPT-5, tanto las métricas de precisión como de sensibilidad mejoraron en comparación con el paradigma de aplicación RAG. La precisión promedio aumentó en 36% y la sensibilidad en 11%.

Tabla 8: Métricas Prompt

Combinación	Precisión (promedio)	Sensibilidad (promedio)	Puntuación F1
LLM: GPT-5	0.849	0.847	0.848
LLM: Mistral	0.492	0.505	0.498

Como se muestra en la Figura 10, tanto la precisión como la consistencia de los resultados entre iteraciones mejoraron para esta combinación. Los requerimientos para los cuales la precisión del LLM disminuyó significativamente fueron aquellos que involucraban cálculos matemáticos que incluían operaciones más allá de sumas y restas simples, como por ejemplo raíces cuadradas.

Figura 10: Métricas de precisión utilizando prompts con GPT-5



El número de casos de prueba faltantes también disminuyó en comparación con el paradigma de aplicación RAG como se observa en la Figura 11. Aunque la cantidad de casos de prueba faltantes sigue aumentando cuando la complejidad de los requisitos se incrementa (particularmente cuando intervienen más entradas o cálculos de límites), el rendimiento general mejoró. En promedio, incluso los puntajes más bajos obtenidos con este paradigma de aplicación siguen siendo más altos que los alcanzados con el paradigma RAG.

En el caso de Mistral, la precisión disminuyó en 9% y la sensibilidad en 7% en comparación con el paradigma de aplicación RAG. La consistencia siguió siendo un problema, ya que se generaron resultados diferentes en las múltiples iteraciones como se muestra en la Figura 12. El único tipo de requisitos que produjo resultados relativamente consistentes y precisos fue aquel que involucraba únicamente entradas booleanas.

Figura 11: Métricas de sensibilidad utilizando prompts con GPT-5

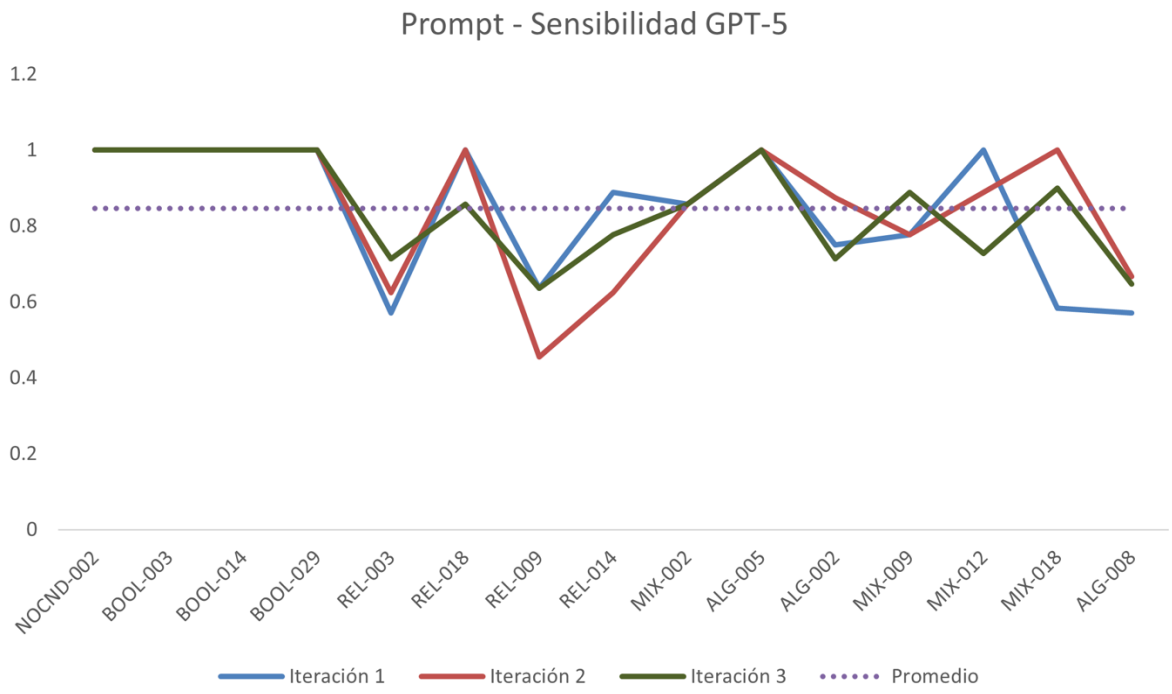


Figura 12: Métricas de precisión utilizando prompts con Mistral

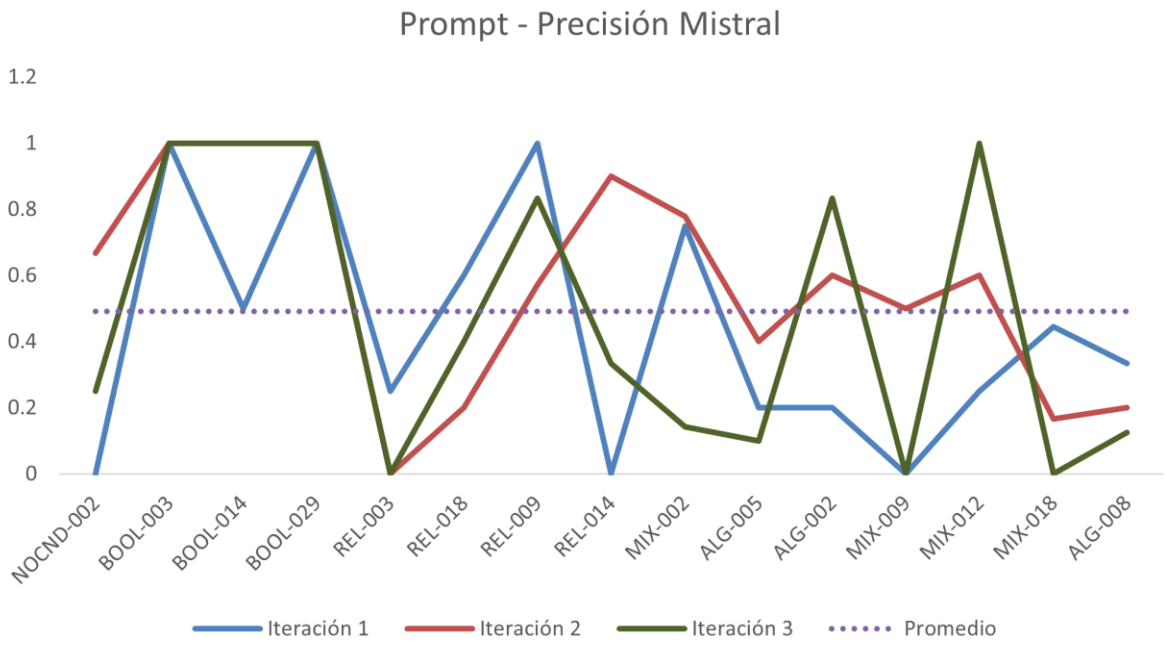
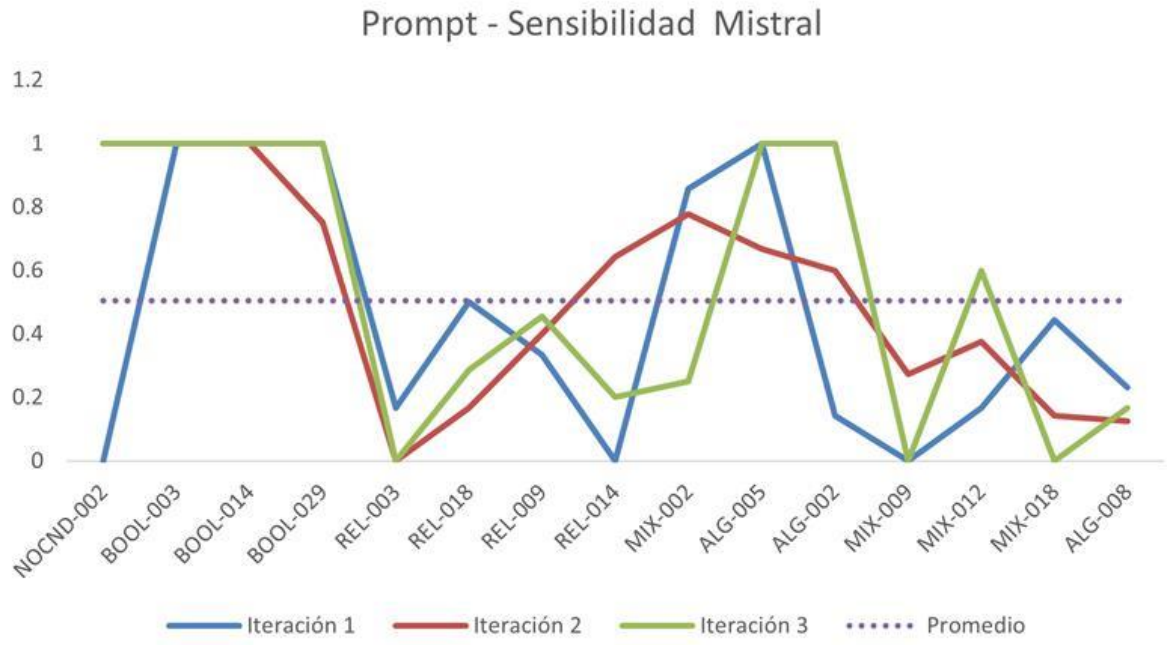


Figura 13: Métricas de sensibilidad utilizando prompts con Mistral



La Tabla 9 muestra el número de casos de prueba faltantes e incorrectos para ambas aplicaciones utilizando *prompts*. En el caso de Mistral, el número de

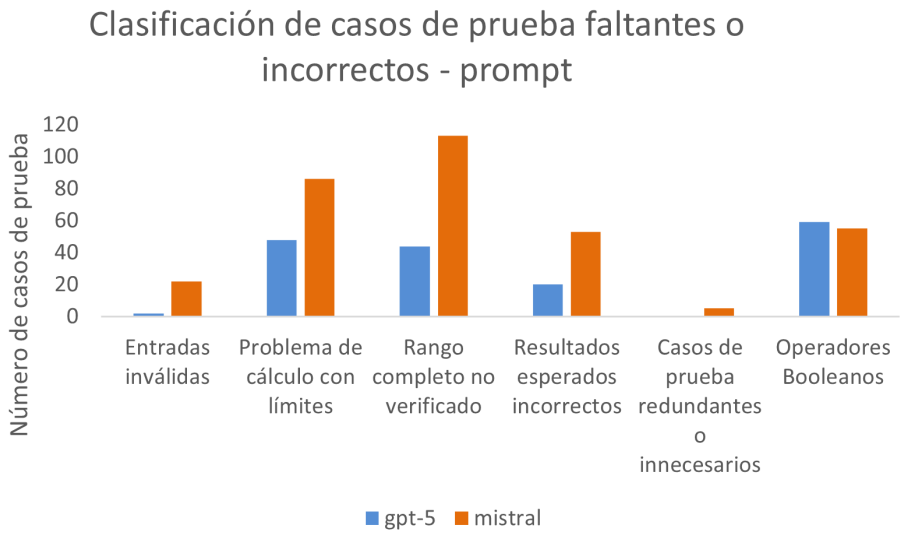
casos de prueba faltantes se mantiene casi igual en comparación con la aplicación RAG, mientras que el número de casos de prueba incorrectos sugeridos aumentó en 27%. En el caso de GPT-5, el número de casos de prueba incorrectos y de casos de prueba faltantes disminuyó en 53% y 26%, respectivamente.

Tabla 9: Casos de prueba faltantes e incorrectos utilizando prompts

Tipo de problema	Prompt GPT-5	Prompt Mistral
Casos de prueba faltantes	69	163
Casos de prueba incorrectos	58	143

Como se muestra en la Figura 14, el principal tipo de problemas identificados utilizando este paradigma de aplicación siguen siendo los casos de prueba faltantes o incorrectos relacionados con límites, cálculos de tolerancia, casos de prueba innecesarios y valores faltantes en los casos de prueba para cubrir todo el rango de la señal (valores máximo, mínimo y medio).

Figura 14: Clasificación de casos de prueba faltantes o incorrectos utilizando prompts



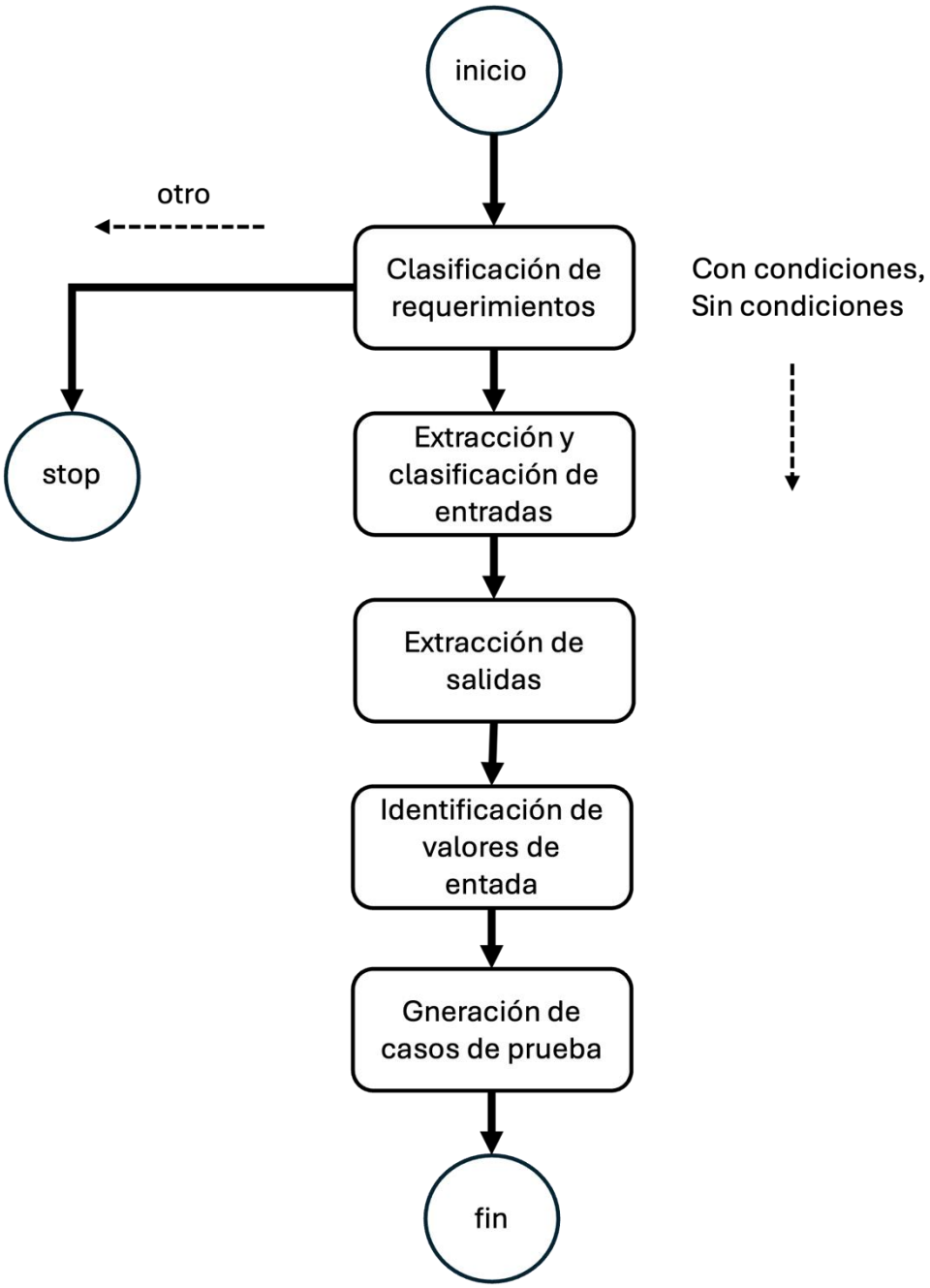
Agregar un procedimiento detallado en el prompt condujo a una mejora en el rendimiento de la combinación con GPT-5, mientras que en Mistral tuvo el efecto contrario. Dado que GPT-5 tiene más parámetros que Mistral, pudo procesar mejor *prompts* más largos para cumplir con las tareas solicitadas por el ingeniero. En contraste, Mistral tuvo un mejor desempeño con *prompts* más cortos, aunque de igual manera generó una cantidad significativa de casos de prueba faltantes e incorrectos.

10.1.3 Agentes

Una forma de utilizar este paradigma de aplicación es diseñando una arquitectura de flujo de trabajo en la que cada nodo sea responsable de una tarea específica (Anthropic, 2024). Esta implementación fue elegida porque proporciona un mayor control sobre las acciones y decisiones realizadas por los LLM; además, permite observar las respuestas generadas por los LLM en cada parte del fujo de trabajo.

La Figura 15 muestra el flujo de trabajo que se implementó para seguir la estrategia de generación de casos de prueba descrita en el Proceso de Identificación de Casos de Prueba. El flujo de trabajo cuenta con un estado interno general con varios atributos para almacenar información relacionada con las entradas, salidas, la descripción del requisito y los casos de prueba.

Figura 15: Flujo de trabajo seguido por el agente



En cada paso se realiza una llamada a un LLM para ejecutar una acción específica requerida para la generación de casos de prueba. El flujo de trabajo comienza recuperando las entradas y salidas del requisito. Luego, los valores de entrada se seleccionan según su clasificación de tipo de entrada. Finalmente, el paso de generación de casos de prueba utiliza los atributos almacenados en el

estado interno para producir los casos de prueba en formato tabular. Se utilizó *LangGraph* como *framework* para implementar este flujo de trabajo.

La Tabla **10** muestra las métricas obtenidas para el paradigma de aplicación basado en agentes. En el caso de GPT-5, la precisión disminuyó un 9 % en comparación con la aplicación basada en prompt, mientras que el recall mejoró en un 5 %. Como se observa en la Figura 16, una de las ventajas de esta aplicación es la consistencia lograda entre iteraciones. Esta mejora se debe a que cada LLM en el flujo de trabajo opera dentro de un alcance más reducido, recibe instrucciones más específicas y utiliza prompts más simples, lo que ayuda a minimizar la variación en la salida. Esto contribuyó a reducir el número de casos de prueba faltantes, como se observa en la Figura 17.

Tabla 10: Métricas utilizando agentes

Combinación	**Precisión (promedio)**	**Sensibilidad (promedio)**	**Puntuación F1**
LLM: GPT-5	0.781	0.895	0.834
LLM: Mistral	0.536	0.621	0.576

Figura 16: Métricas de precisión utilizando agentes con GPT-5

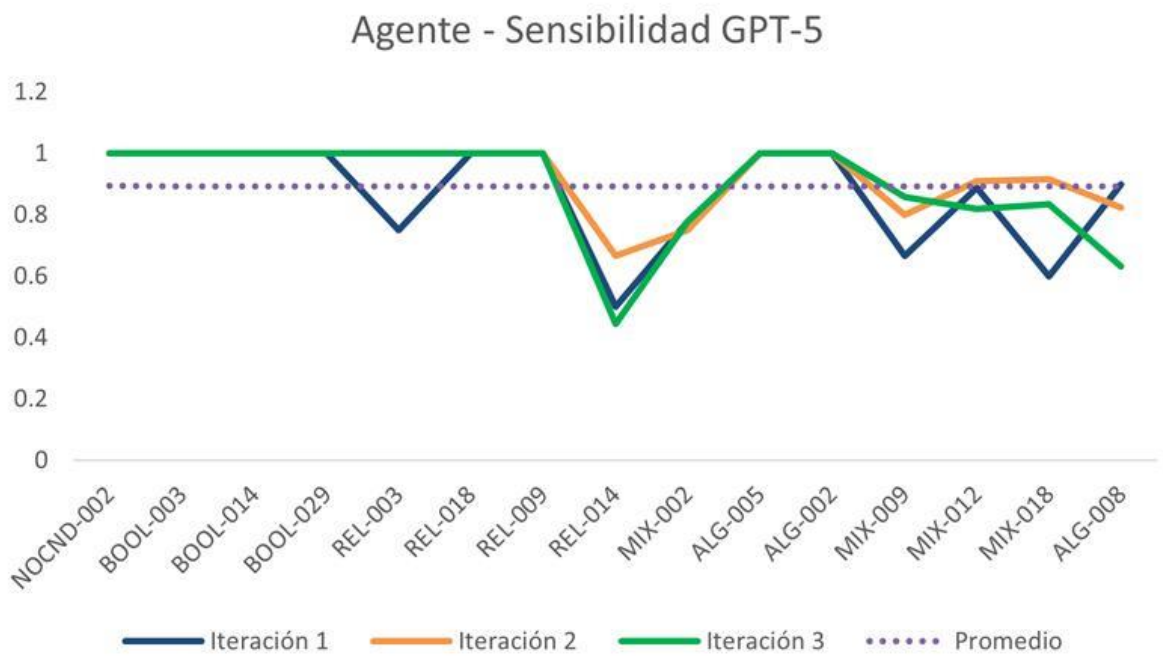
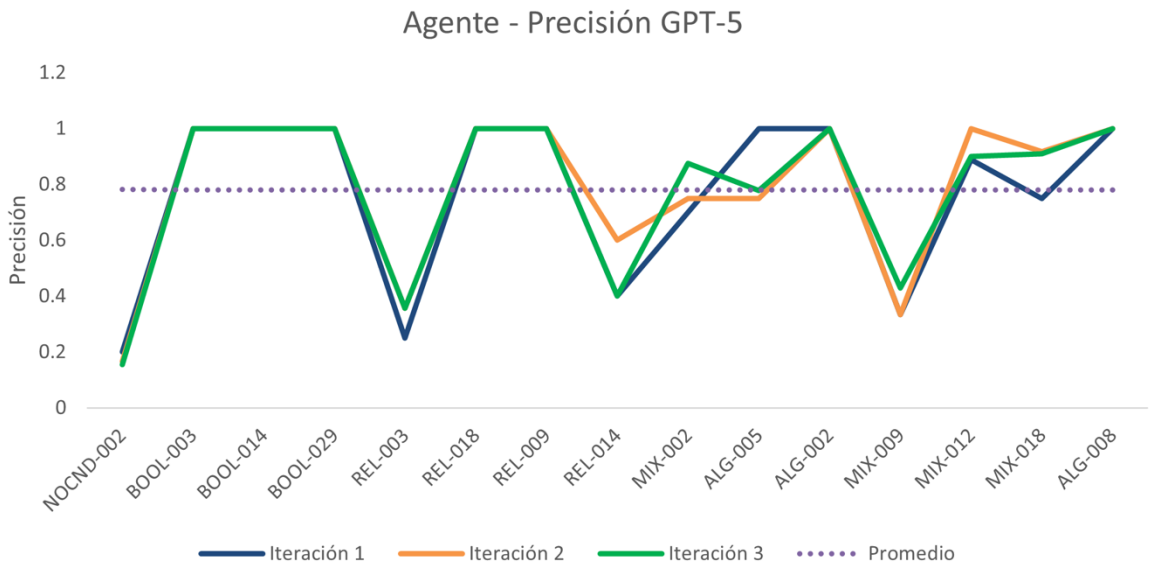


Figura 17: Métricas de sensibilidad utilizando agentes con GPT-5



Por otro lado, dado que el flujo de trabajo implica un encadenamiento de prompts entre diferentes llamadas al LLM, los errores introducidos en un paso se propagan a los pasos posteriores. Una mala clasificación del tipo de requerimiento puede conducir a estrategias de prueba diseñadas para otros tipos de requerimiento, lo que resulta en casos de prueba incorrectos por ejemplo. Si las

entradas no se extraen correctamente, esto generará casos de prueba faltantes en otros pasos. En la Figura 16 se puede observar que la puntuación de precisión más baja obtenida fue para NOCND-002. Esto se debe a que el flujo de trabajo clasificó incorrectamente el tipo de requisito, agregando casos de prueba incorrectos e innecesarios para este tipo de requerimiento.

En el caso de Mistral, la métrica F1 de la Tabla **10** indica que los resultados de precisión y recall mejoraron en comparación con los otros dos paradigmas de aplicación. La precisión se mantiene igual en comparación con los resultados de RAG, que fue la puntuación más alta obtenida en los otros dos paradigmas de aplicación; el recall mejoró en un 15 %.

De manera similar a la combinación con GPT-5, la consistencia de los resultados entre iteraciones mejoró, lo que llevó a menos casos de prueba faltantes y a un mayor número de casos correctos. Como se muestra en la Figura 18 y Figura 19, este paradigma de aplicación aún omite más casos de prueba a medida que aumenta la complejidad de los requisitos. Sin embargo, fue capaz de sugerir algunos casos de prueba que no fueron identificados utilizando los paradigmas de aplicación anteriores.

Figura 18: Métricas de precisión usando agentes con Mistral

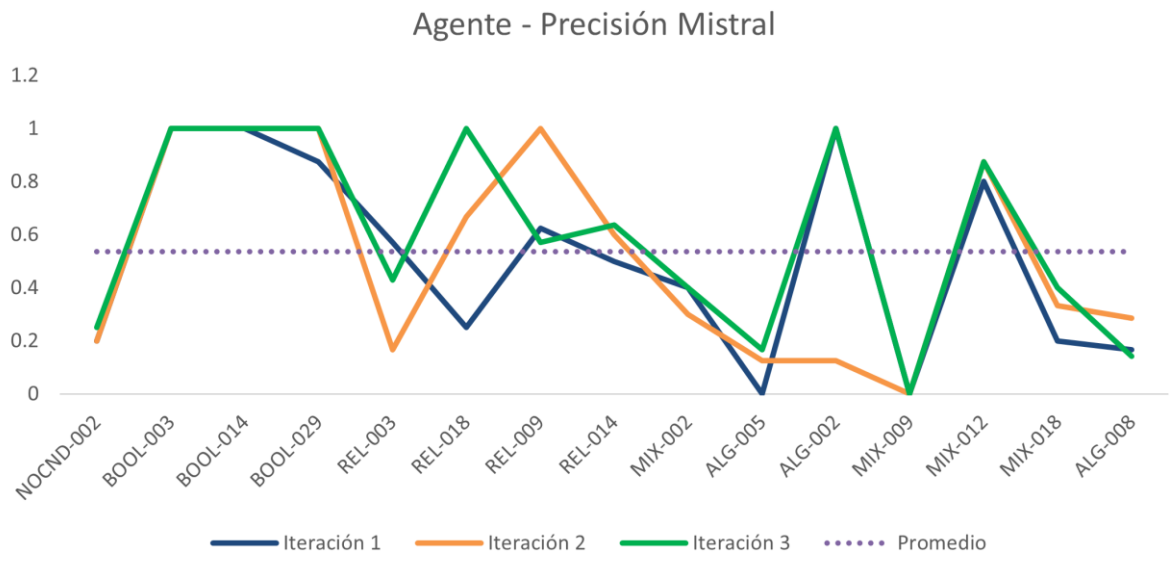
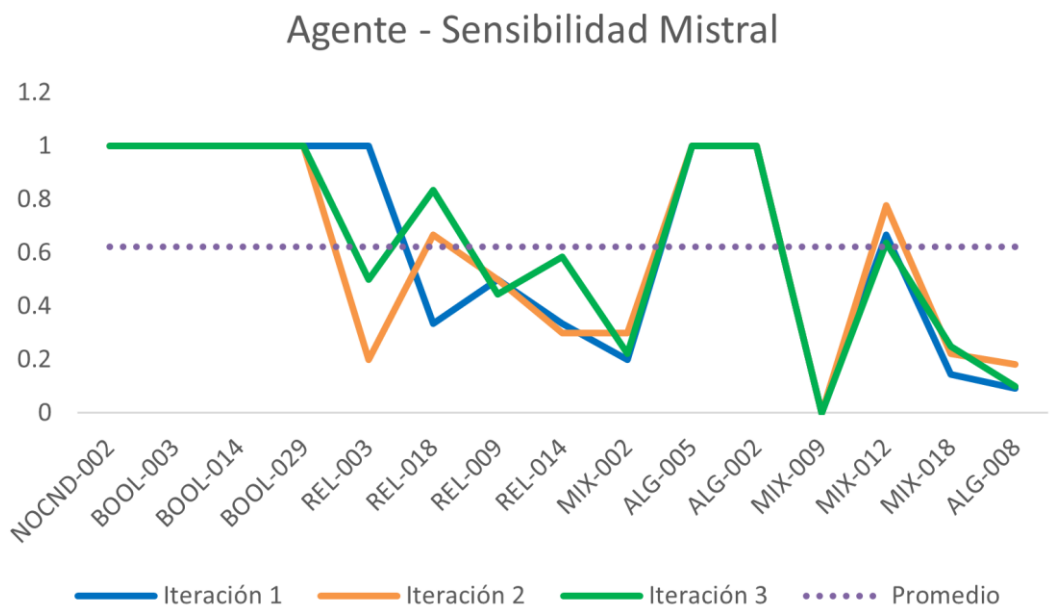


Figura 19: Métricas de sensibilidad usando agentes con Mistral



Se observa una tendencia similar con la métrica de precisión: las puntuaciones más bajas ocurrieron cuando los requisitos involucraban operaciones matemáticas y cálculos de valores límite.

La Tabla 11 muestra el número de casos de prueba faltantes e incorrectos evaluados al utilizar este paradigma de aplicación. En el caso de GPT-5, el problema más común fueron las entradas inválidas, seguido por los cálculos de valores límite. En el caso de las entradas inválidas, este problema fue introducido por una clasificación incorrecta del tipo de requisito, lo que condujo a una estrategia de pruebas incorrecta. Además, los casos de prueba de valores límite cuando existen tolerancias involucradas no siempre se calcularon correctamente.

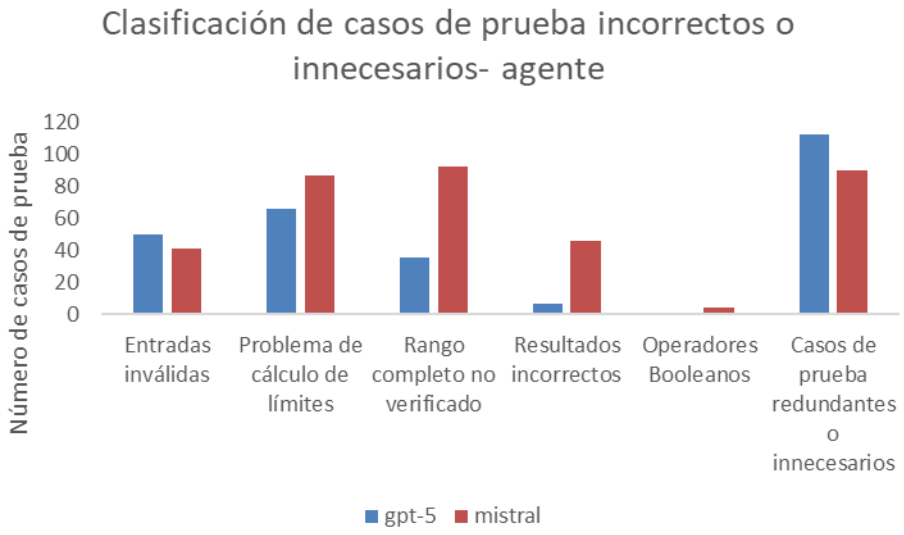
Tabla 11: Casos de prueba faltantes e incorrectos para el paradigma de aplicación con agentes

Tipo de error	Agente GPT-5	Agente Mistral
Casos de prueba faltantes	46	143
Casos de prueba incorrectos	114	132

En el caso de Mistral, los problemas más comunes para este paradigma de aplicación fueron los cálculos de valores límite, los casos de prueba repetidos o innecesarios, y la selección de valores de entrada para cubrir el rango de la señal (valores mínimo, máximo y medio), como se muestra en la Figura 20. En el caso de GPT-5, los problemas más comunes fueron la repetición o la inutilidad de pruebas, valores límite no probados correctamente y entradas inválidas.

La diferencia entre Mistral y GPT-5 en términos de casos de prueba con resultados incorrectos consiste principalmente en los errores matemáticos producidos por el modelo Mistral. En varios casos, este modelo no logró proporcionar un resultado o generó un resultado incorrecto. La diferencia en el rendimiento se correlaciona en parte con la diferencia en el número de parámetros entre ambos modelos, lo que afecta su capacidad para manejar operaciones complejas.

Figura 20: Clasificación de casos de prueba faltantes o incorrectos utilizando agentes



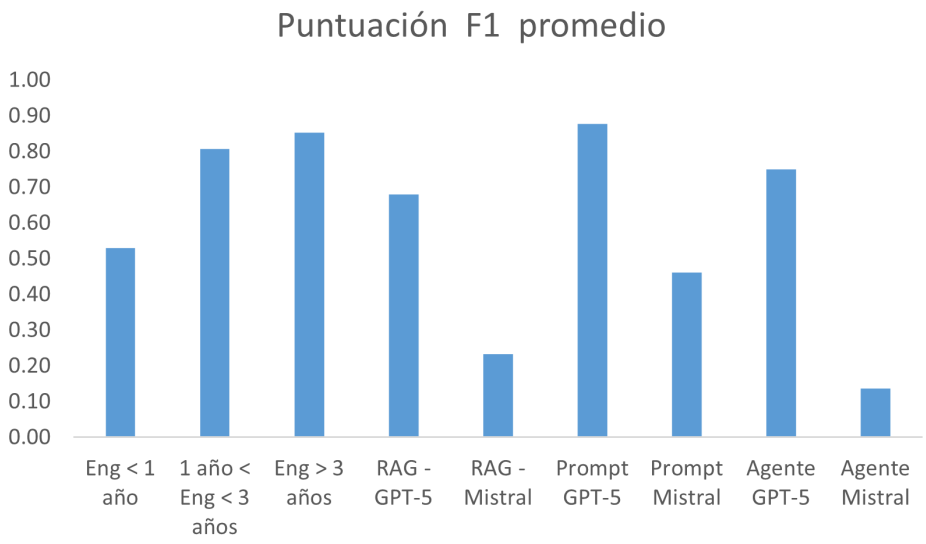
10.2 Experimento Humano – LLMs

La segunda parte de este estudio consiste en evaluar los casos de prueba sugeridos por los paradigmas de aplicación de LLM frente a aquellos generados por ingenieros de verificación con diferentes niveles de experiencia.

Se aplicó una prueba a un grupo de ingenieros de verificación a quienes se les pidió identificar los casos de prueba para tres requisitos diferentes. Se les indicó que proporcionaran los casos de prueba en formato tabular para facilitar su visualización y revisión, como también se describe en la sección de evaluación de LLM. Posteriormente, se instruyó a los paradigmas de aplicación de LLM para que identificaran y generaran casos de prueba a partir de los mismos requisitos.

Los resultados se muestran en la Figura 21. Los ingenieros fueron agrupados según su nivel de experiencia y evaluados tanto mediante la métrica F1 (que evalúa precisión y recall) como por el tiempo necesario para identificar los casos de prueba. Los resultados indican que la experiencia es un factor importante que influye en la calidad de los casos de prueba, ya que los ingenieros con tres o más años de experiencia obtuvieron puntuaciones más altas que los otros grupos.

Figura 21: Puntuaciones F1



Los errores más comunes observados en los casos de prueba elaborados por humanos incluyeron casos de prueba faltantes para verificar los valores máximos y mínimos del rango de entrada, así como casos de prueba incorrectos o faltantes para verificar valores límite, debido a que las tolerancias y resoluciones no se calcularon correctamente.

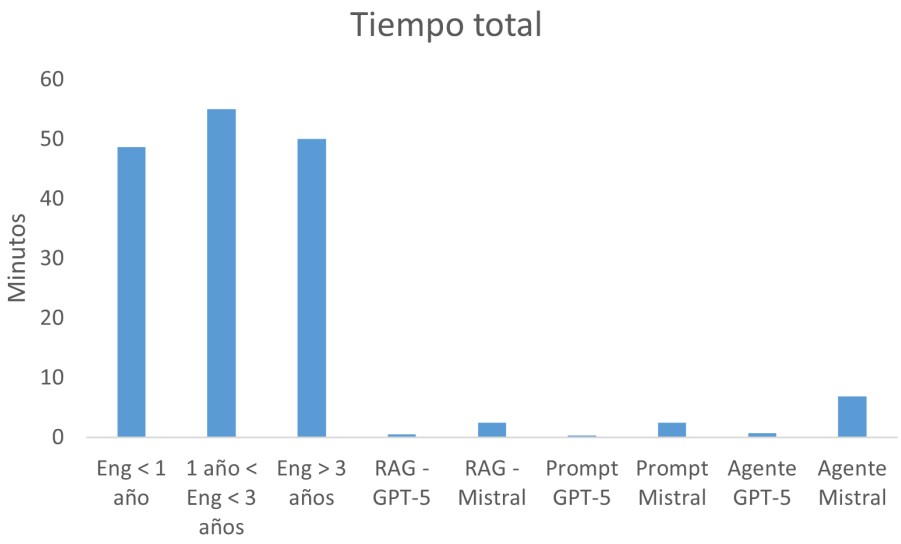
En el caso de los paradigmas de aplicación de LLM, los resultados fueron consistentes con los análisis realizados en secciones anteriores, donde los paradigmas de prompt y agentes con GPT-5 obtuvieron las puntuaciones más altas.

Además de las altas puntuaciones obtenidas al utilizar estos paradigmas de aplicación, la Figura 22 muestra que el tiempo que les tomó a los ingenieros generar los casos de prueba con soporte de LLM se redujo considerablemente. Los ingenieros pudieron identificar casos de prueba en menos de 1 minuto, en comparación con los 50 minutos o más requeridos al completar la tarea sin herramientas basadas en LLM.

No obstante, es importante señalar que los tiempos medidos reflejan únicamente el paso inicial de generación. La salida generada por el LLM puede requerir una evaluación y refinamiento posterior por parte de un humano, los cuales

no fueron incluidos en la medición del tiempo. Por lo tanto, las aparentes ganancias de rendimiento representan un límite superior y no una medición completa del esfuerzo total requerido.

Figura 22: Tiempo total para la generación de casos de prueba



Por otro lado, la combinación que produjo las puntuaciones más bajas fue la aplicación basada en agentes utilizando Mistral. Debido a que este enfoque se basa en múltiples *prompts* encadenados, los errores introducidos en un paso se propagaron a los pasos posteriores.

Además, el LLM no siguió de manera consistente todas las instrucciones proporcionadas en los *prompts* de cada etapa. El modelo Mistral mostró un mejor desempeño cuando los prompts eran más cortos y contenían menos detalles.

11 Capítulo 11. Guía de integración de modelos LLMs en procesos de identificación de casos de prueba

11.1 Introducción

La industria aeroespacial requiere niveles altos de seguridad, trazabilidad y confiabilidad. La incorporación de inteligencia artificial introduce nuevos desafíos

relacionados con el uso de herramientas para la validación y verificación de los sistemas.

Debido a la naturaleza probabilística de la inteligencia artificial, los resultados generados por este tipo de herramientas pueden incluir errores o incluso omitir la identificación de problemas existentes en los artefactos que el ingeniero utiliza para generar casos de prueba. Esto también implica dificultades para calificar este tipo de herramientas, ya que no se puede obtener un resultado determinístico.

Esta guía define un proceso estructurado para la elaboración de herramientas que permitan generar de casos de prueba utilizando inteligencia artificial. Este proceso es adaptable a distintos tipos de requerimientos y está alineado con principios de seguridad de la industria aeroespacial tales como la guía DO-178C (RTCA, 2011) y la hoja de ruta propuesta por la FAA para el uso de inteligencia artificial en sistemas de aviación. (Administration, 2024).

11.2 Objetivo de la guía

Desarrollar una guía estructurada para la generación de casos de prueba a partir de requerimientos de software aeronáutico expresados en lenguaje natural, orientada a dos propósitos complementarios:

- Objetivo principal: Definir un proceso reproducible y trazable para la identificación de casos de prueba que sirva como base para el desarrollo de herramientas asistidas por Inteligencia Artificial en la industria aeroespacial. La guía pretende documentar el proceso que permite generar, para un requerimiento dado, un conjunto de casos de prueba que satisfaga los criterios de verificación establecidos en el DO-178C y que pueda ser validado por un ingeniero de verificación calificado.
- Objetivo secundario: Proporcionar un material de referencia para el entrenamiento de personal de verificación en el uso responsable de herramientas impulsadas por IA, estableciendo criterios claros sobre el rol del ingeniero como responsable final de la aprobación de los casos de prueba generados.

El alcance de esta guía cubre requerimientos de software para sistemas aeronáuticos clasificados bajo los niveles de criticidad DAL A según la definición del estándar DO-178C (RTCA, 2011). Se busca cumplir con los niveles de criticidad más altos para cubrir consecuentemente con niveles más bajos.

Asimismo, la guía se limita a requerimientos expresados en lenguaje natural, excluyendo aquellos que se apoyan en modelos formales, diagramas de máquinas de estado o especificaciones gráficas, los cuales requieren un tratamiento metodológico diferente.

11.3 Principios Fundamentales Basados en la Hoja de Ruta de la FAA

El diseño de herramientas impulsadas por IA para la generación de casos de prueba debe estar guiado por un conjunto de principios que aseguren que la calidad, trazabilidad y seguridad de los procesos de verificación no se vean comprometidos al introducir este tipo de tecnología. La hoja de ruta publicada por la FAA en 2024 (Administration, Roadmap for artificial intelligence safety assurance – Version I, 2024) establece siete principios fundamentales para el uso de IA en sistemas de aviación. A continuación, se describe cada principio y su aplicación concreta al contexto de generación de casos de prueba.

11.3.1 Trabajar bajo el ecosistema existente de aviación

Este principio establece que toda herramienta o proceso que incorpore IA debe ser compatible con los planes y estándares vigentes de la industria de aviación, sin reemplazarlos ni contradecirlos.

En el contexto de la verificación de software aeronáutico, el DO-178C es el estándar de referencia. Dicha guía establece que los casos de prueba deben derivarse de los requerimientos del sistema y no del código fuente, ya que este enfoque es el más efectivo para detectar errores tanto en el código como en los propios requerimientos. Por esta razón, las herramientas desarrolladas deben garantizar que los casos de prueba sugeridos se originen exclusivamente en el análisis de los requerimientos.

Adicionalmente, el DO-178C en su sección 12, complementado por el estándar DO-330 (RTCA, DO-330: Software Tool Qualification Considerations, 2011), establece que toda herramienta que elimine, reduzca o automatice una actividad de verificación que de otro modo requeriría revisión humana debe ser formalmente calificada. Dado que los modelos de IA generativos son sistemas no determinísticos no es posible garantizar repetibilidad y, por lo tanto, no cumplen con los criterios requeridos para la calificación de herramientas bajo DO-330. Por esta razón, se establece como principio de diseño que las herramientas desarrolladas deben asistir al ingeniero de verificación y no automatizar completamente la generación de casos de prueba. La decisión final sobre la aceptación, rechazo o modificación de los casos de prueba sugeridos recae siempre en el ingeniero responsable.

11.3.2 Diseñar funcionalidades para incrementar los niveles de seguridad

Las herramientas impulsadas por IA deben ser diseñadas con el propósito explícito de mejorar la calidad y seguridad de los procesos existentes, no únicamente de reducir tiempos de ejecución.

En este caso, la funcionalidad principal de asistencia en la identificación de casos de prueba aporta valor en dos dimensiones. En primer lugar, permite que ingenieros con mayor experiencia optimicen su tiempo al contar con una propuesta inicial que pueden revisar y refinar. En segundo lugar, apoya a ingenieros con menor experiencia al ofrecerles una guía estructurada basada en criterios establecidos, reduciendo la probabilidad de que se omitan casos de prueba relevantes, como condiciones límite o clases de equivalencia fuera de rango.

11.3.3 Evitar la personificación de sistemas de inteligencia artificial.

Las herramientas impulsadas por IA no deben ser diseñadas para tomar decisiones ni ejecutar acciones que son responsabilidad de un profesional humano dentro del proceso de verificación.

En consecuencia, el flujo de trabajo de cualquier herramienta desarrollada bajo esta guía debe incluir obligatoriamente una etapa de revisión humana, en la cual el ingeniero de verificación aprueba, rechaza o corrige cada caso de prueba

sugerido antes de que este sea incorporado al plan de pruebas formal. Ningún caso de prueba debe avanzar en el proceso sin haber sido revisado y autorizado explícitamente por un ingeniero calificado.

11.3.4 Diferenciar entre sistemas aprendidos y sistemas que aprenden.

La hoja de ruta de la FAA distingue dos categorías de sistemas de IA según su comportamiento durante la operación.

Los sistemas aprendidos son aquellos entrenados con un conjunto de datos estático y congelado. Una vez completado el entrenamiento, su comportamiento es fijo y predecible dentro de los rangos de operación definidos. Estos sistemas pueden ser evaluados, documentados y sometidos a procesos de verificación convencionales. Los Grandes Modelos de Lenguaje (LLMs) de uso general, como los que podrían utilizarse como base para las herramientas descritas en esta guía, pertenecen a esta categoría siempre que se utilicen en modo de inferencia sin modificación de sus pesos durante la operación.

Los sistemas que aprenden son aquellos capaces de modificar su comportamiento durante la fase de operación, generalmente mediante técnicas de aprendizaje por refuerzo u otros mecanismos de actualización continua. Este tipo de sistemas no cuenta aún con aprobación regulatoria para su uso en aviación, dado que su comportamiento futuro no puede predecirse ni acotarse con certeza.

A partir de esta distinción, se establecen los siguientes criterios de diseño para las herramientas desarrolladas bajo esta guía:

- Queda prohibido incorporar mecanismos de aprendizaje activo durante la operación de la herramienta, como el ajuste de parámetros basado en la retroalimentación del usuario en tiempo real. Esto podría llevar a la herramienta a reforzar prácticas incorrectas y compromete la estabilidad y mantenibilidad de cada versión.
- Cada versión de la herramienta debe ser evaluada mediante benchmarks documentados antes de su despliegue en producción, permitiendo comparar su desempeño entre versiones y detectar regresiones.

- La revisión humana descrita en el Principio 3 actúa también como mecanismo de contención ante cualquier error introducido por el modelo.

11.3.5 Adoptar enfoques incrementales para la implementación

Este principio establece que las capacidades de las herramientas de IA deben ampliarse de forma gradual y controlada, aprendiendo de las limitaciones y errores identificados en cada etapa antes de incrementar la complejidad o el alcance de las funcionalidades.

Aplicado a esta guía, se recomienda que el desarrollo de herramientas comience por la asistencia en la identificación de casos de prueba antes de extenderse hacia la generación automática de scripts de prueba o la ejecución y análisis de resultados, que son actividades de mayor criticidad y complejidad. Cada nueva capacidad incorporada debe estar respaldada por evidencia documentada del correcto funcionamiento de la capacidad anterior.

11.3.6 Buscar la mejora continua de la seguridad

Al igual que el principio anterior, la herramienta debe ser utilizada primero con funcionalidades de menor riesgo e impacto antes de ser utilizadas para labores más complejas y de mayor riesgo.

Los errores identificados por los ingenieros durante la etapa de revisión humana (casos de prueba incorrectos, omisiones sistemáticas, malinterpretaciones de condiciones límite) deben registrarse formalmente y analizarse para determinar si requieren ajustes en el proceso, en el entrenamiento del modelo o en los criterios de evaluación definidos en la guía.

11.3.7 Utilizar estándares de la industria para asegurar cumplimiento normativo

Se deben adoptar los estándares desarrollados por la industria de aviación con respecto al uso de inteligencia artificial

11.4 Descripción del proceso

Esta sección describe el proceso recomendado para generar casos de prueba a partir de requerimientos de software aeronáutico expresados en lenguaje natural. El proceso está diseñado para ser seguido tanto por ingenieros de verificación como por herramientas asistidas por inteligencia artificial, y en ambos casos la aprobación final de los casos de prueba generados recae en un ingeniero calificado.

El proceso se compone de tres etapas secuenciales: clasificación del requerimiento, análisis de variables y generación de casos de prueba.

11.5 Clasificación del requerimiento

Como primer paso del proceso, se recomienda clasificar el requerimiento en diferentes categorías. Esto permite analizar cuáles criterios de evaluación aplican para los casos de prueba generados.

Los tipos de requerimientos que se proponen son:

- Requerimientos con condiciones: son aquellos en los que el comportamiento del sistema depende del estado de una o más variables de entrada. Es decir, el sistema ejecutará una acción determinada únicamente si se cumplen las condiciones especificadas en el requerimiento. Estos requerimientos típicamente contienen construcciones lógicas del tipo"si... entonces...","cuando... el sistema *deberá..."*.
- Requerimientos sin condiciones: son aquellos en los que el sistema ejecuta una acción de forma independiente al estado de las variables de entrada, o bien en los que no existen variables de entrada que determinen si la acción ocurre o no. En este tipo de requerimiento, la verificación se centra en confirmar que la acción siempre se ejecuta correctamente y que los valores producidos son los esperados. Este tipo de requerimiento puede incluso no tener variables de entrada.
- Requerimientos con imágenes: requerimientos que se apoyan de imágenes para describir el comportamiento esperado del sistema.

- Requerimientos basados en máquinas de estado: requerimiento que utiliza un modelo de máquina de estado para describir el comportamiento del sistema. Usualmente, se utiliza un conjunto de requerimientos para describir las múltiples condiciones y transiciones de la máquina de estado.

Cabe destacar que en este estudio, se consideró únicamente los requerimientos que utilizan lenguaje natural para describir el comportamiento del sistema. Además, los requerimientos que incluyen imágenes o describen máquinas de estado no son parte del enfoque de esta investigación.

11.6 Análisis del requerimiento

Una vez clasificado el requerimiento, se procede a analizar las variables de este con el objetivo de proveer una mejor guía con respecto a los criterios de evaluación que aplican para el requerimiento a verificar.

Existen tres tipos de variables: condiciones iniciales, variables de entrada y variables de salida.

Las condiciones iniciales corresponden a las variables o requisitos necesarios para poder verificar el comportamiento descrito en el requerimiento.

Las condiciones de entrada representan las variables definidas en un requerimiento que determinan si deben ocurrir acciones o comportamientos específicos del sistema. También, en caso de no haber condiciones en el requerimiento, este tipo de variable puede estar presente en las acciones que el sistema va a ejecutar; por ejemplo, si el sistema debe enviar un mensaje que contiene el voltaje del sistema, el voltaje se considera una variable de entrada el cual debe ser modificado en varios casos de probado con el fin de verificar que el valor es actualizado correctamente.

Las variables de entrada se dividen en tres tipos de acuerdo con el análisis realizado en el Capítulo 7. Clasificación de requerimientos: booleanas, de rango continuo y enumeración.

Las variables de salida corresponden a las acciones o comportamientos esperados del sistema y contienen los valores esperados a verificar en los casos de prueba.

11.7 Generación de casos de prueba

Una vez clasificados los requerimientos y haber procesado las variables, se recomienda utilizar la siguiente guía para definir los valores a probar para cada entrada:

Tabla 12: Criterio de evaluación de casos de prueba según el tipo de requerimiento y variable de entrada

Criterio ID	Criterio de evaluación	Tipo de requerimiento	Tipo de variable de entrada
CI-1	Los casos de prueba de verificación incluyen representaciones de clases de equivalencia para las entradas.	Con condiciones	Rango continuo Enumeración Booleano
CI-2	Los casos de prueba verifican correctamente los límites.	Con condiciones	Rango continuo
CI-3	Los casos de prueba verifican correctamente los valores mínimos y máximos del rango.	Con condiciones Sin condiciones	Rango continuo
CI-4	Los casos de prueba toman en consideración aspectos relacionados con variables que involucran tiempo.	Con condiciones	Rango continuo
CI-5	Los casos de prueba cubren operaciones Booleanas.	Con condiciones	Booleano
CI-6	Los casos de prueba incluyen el	Con	Rango

Criterio ID	Criterio de evaluación	Tipo de requerimiento	Tipo de variable de entrada
	resultado esperado.	condiciones	continuo
		Sin condiciones	Booleano Enumeración
CI-7	Cuando es posible los casos de prueba incluyen representaciones de clases de equivalencia para entradas fuera de los límites.	Con condiciones	Rango continuo
CI-8	Los casos de prueba predicen correctamente los resultados esperados basándose en la lógica descrita en el requerimiento y en los valores de entrada especificados en cada caso de prueba.	Con condiciones	Rango continuo
		Sin condiciones	Booleano Enumeración

Una vez seleccionados los valores necesarios para cumplir con los criterios de evaluación, se procede a generar los casos de prueba. Los casos de prueba deben tener un identificador del caso de prueba, identificador del requerimiento, descripción, tipo de prueba (normal o robusta), valores esperados, valores de entrada y condiciones iniciales.

El método recomendado para combinar los valores de las variables de entrada en los casos de prueba es el análisis de una variable a la vez (Library, 2026). Este método consiste en seleccionar una variable de entrada y generar casos de prueba que cubran todos sus valores relevantes mientras las demás variables de entrada se mantienen en un valor fijo.

El valor fijo asignado a las variables que no están siendo probadas debe seleccionarse de manera que se cumpla lo siguiente:

- Las variables fijas satisfacen las condiciones del requerimiento, de modo que sea la variable que se está probando la que determine el valor de la variable de salida.
- El valor fijo corresponde a un valor representativo y documentado, preferiblemente el valor nominal o central del rango de operación de esa variable.

Una vez completada la verificación de todos los valores de una variable, se repite el proceso con la siguiente variable de entrada, manteniendo las demás fijas.

A continuación, se presenta un ejemplo de aplicación del formato para el requerimiento: "Si la temperatura del motor supera los 120°C, el sistema deberá activar la alarma de sobrecalentamiento."

Tabla 13: Formato de casos de prueba

ID	Requerimiento	Descripción	Tipo	Condiciones Iniciales	Variables de Entrada	Resultado Esperado
TC-01	REQ-001	Verificar que la alarma no se activa cuando la temperatura está por debajo del límite	Normal	Sistema de alarmas inicializado y operativo	Temperatura: 100°C	Alarma: Inactiva

11.8 Identificación de Riesgos

El uso de herramientas asistidas por inteligencia artificial en el proceso de generación de casos de prueba introduce riesgos que deben ser identificados, evaluados y mitigados antes de que dichas herramientas sean utilizadas en proyectos de software aeronáutico. Esta sección presenta los riesgos identificados, organizados según su origen, y define para cada uno su descripción, su impacto

potencial sobre el proceso de verificación y las medidas de mitigación recomendadas.

Los riesgos se organizan en cuatro categorías: riesgos relacionados con la calidad de los casos de prueba generados, riesgos relacionados con el modelo de Inteligencia Artificial, riesgos relacionados con el proceso y riesgos relacionados con el cumplimiento normativo.

11.8.1 Riesgos relacionados con la calidad de los casos de prueba

Las siguientes secciones describen riesgos desde los puntos de vista técnico, de procesos y de uso que deben considerarse al implementar herramientas impulsadas por IA para generar casos de prueba basados en los requerimientos.

11.8.1.1 Casos de prueba faltantes

La herramienta puede omitir casos de prueba necesarios para satisfacer uno o más criterios de evaluación definidos en la Tabla 2. Esto puede ocurrir cuando el modelo de IA no identifica correctamente todas las clases de equivalencia, no genera todos los casos necesarios para cumplir con los límites indicados, o no detecta variables de entrada con componentes temporales.

Una cobertura de pruebas incompleta puede resultar en que defectos presentes en el software no sean detectados durante la fase de verificación, comprometiendo directamente los objetivos de seguridad establecidos por el DO-178C.

El ingeniero de verificación debe utilizar la Tabla 2 como lista de verificación explícita al revisar los casos de prueba generados, confirmando que cada criterio aplicable al tipo de requerimiento y tipo de variable ha sido satisfecho. Adicionalmente, se recomienda que la herramienta incluya una función de autoverificación que indique al usuario qué criterios se cubrieron y cuáles no, facilitando la revisión.

11.8.1.2 Casos de prueba con resultado esperado incorrecto

La herramienta puede generar casos de prueba con valores de entrada válidos pero con un resultado esperado que no es consistente con la lógica del requerimiento. Este error es particularmente peligroso porque el caso de prueba aparenta ser completo y correcto, pero en realidad validaría un comportamiento incorrecto del sistema.

Si un caso de prueba con resultado esperado incorrecto es aprobado sin revisión adecuada, el proceso de verificación podría concluir que el software cumple con el requerimiento cuando en realidad no lo hace, o viceversa.

El criterio CI-8 definido en la Tabla 12 actúa como mecanismo de control para este riesgo, exigiendo que el ingeniero verifique explícitamente la consistencia lógica entre los valores de entrada y el resultado esperado en cada caso de prueba. Se recomienda que esta verificación sea obligatoria y que quede documentada como parte del registro de revisión.

11.8.1.3 Casos de prueba duplicados o redundantes

La herramienta puede generar múltiples casos de prueba que verifican exactamente el mismo aspecto del requerimiento con valores equivalentes, sin aportar cobertura adicional. Aunque este riesgo no compromete directamente la seguridad, incrementa el tiempo y costo de ejecución de las pruebas sin beneficio proporcional.

El riesgo radica en el incremento injustificado del esfuerzo de verificación para mantener casos de prueba que no brindan valor adicional.

El ingeniero de verificación debe identificar y eliminar casos de prueba redundantes durante la revisión, verificando que cada caso de prueba aporte evidencia única con respecto a los criterios definidos en la Tabla 2.

11.8.1.4 Condiciones límite no evaluadas correctamente

El tratamiento de condiciones límite es uno de los aspectos más complejos de la generación de casos de prueba y uno de los más propensos a errores. La herramienta puede identificar incorrectamente el valor límite, confundir si el límite es inclusivo o exclusivo, o no generar casos de prueba para ambos lados del límite.

Una verificación incorrecta de estos valores puede dejar sin detectar errores de lógica en el código que solo se manifiestan en valores específicos cercanos a los límites definidos en los requerimientos.

El criterio CI-2 debe aplicarse con especial rigor durante la revisión humana. Se recomienda que el ingeniero verifique explícitamente, para cada condición límite identificada en el requerimiento, que existen casos de prueba para el valor exacto del límite, para un valor inmediatamente por debajo y para un valor inmediatamente por encima, y que el resultado esperado de cada uno es consistente con la lógica del requerimiento, prestando atención a si el operador de comparación es estricto (>) o no estricto (≥).

11.8.2 Riesgos Relacionados con el Modelo de Inteligencia Artificial

Las siguientes secciones se enfocan en describir posibles riesgos relacionados con el uso de herramientas impulsadas por IA debido a su naturaleza probabilística.

11.8.2.1 Interpretación incorrecta del requerimiento

Los modelos LLM pueden interpretar incorrectamente requerimientos ambiguos, con estructura gramatical compleja, o que contienen términos técnicos específicos del dominio aeronáutico que no estaban suficientemente representados en el entrenamiento del modelo. Esta interpretación errónea puede propagarse a todos los casos de prueba generados para ese requerimiento.

Si la herramienta malinterpreta el comportamiento descrito en el requerimiento, la totalidad de los casos de prueba generados puede ser incorrecta o

irrelevante, requiriendo su descarte completo y la intervención del ingeniero para generar los casos de prueba manualmente.

Se recomienda que la herramienta presente al usuario su interpretación del requerimiento, incluyendo las variables identificadas y su clasificación, antes de generar los casos de prueba, permitiendo al ingeniero detectar y corregir errores de interpretación en una etapa temprana del proceso. Adicionalmente, los requerimientos que presenten ambigüedad deben ser reportados por la herramienta para su clarificación con el autor antes de proceder.

11.8.2.2 Variabilidad no determinística de los resultados

Como se mencionó en la sección Conceptos Fundamentales, los modelos de IA generativos son no determinísticos. Esto significa que ante el mismo requerimiento, la herramienta puede producir conjuntos de casos de prueba diferentes en distintas ejecuciones, lo que dificulta la reproducibilidad y el mantenimiento del proceso de verificación.

La falta de reproducibilidad es incompatible con los requisitos de trazabilidad y consistencia del DO-178C. Si los casos de prueba generados no son reproducibles, no es posible demostrar que el proceso de verificación fue aplicado de manera consistente a lo largo del proyecto.

Los casos de prueba aprobados por el ingeniero deben ser almacenados y gestionados bajo control de configuración, de modo que la versión aprobada quede registrada independientemente de lo que la herramienta pudiera generar en una ejecución futura. La herramienta debe ser tratada como un asistente generador de propuestas, no como la fuente definitiva de los casos de prueba.

11.8.2.3 Degradación del rendimiento entre versiones del modelo

Las actualizaciones al modelo de IA subyacente pueden modificar su comportamiento de formas no previstas, resultando en que una versión actualizada del modelo produzca resultados de menor calidad que la versión anterior para ciertos tipos de requerimientos.

Si la degradación no es detectada antes de que la nueva versión sea utilizada en producción, los casos de prueba generados durante ese período pueden ser de menor calidad sin que el equipo de verificación sea consciente de ello.

Cada nueva versión de la herramienta debe ser evaluada mediante un conjunto de benchmarks documentados antes de su despliegue, tal como se establece en el Principio Diferenciar entre sistemas aprendidos y sistemas que aprenden.. Estos benchmarks deben incluir requerimientos representativos de cada tipo definido en la sección Clasificación del requerimiento, con conjuntos de casos de prueba de referencia aprobados por ingenieros expertos, contra los cuales se pueda medir el desempeño de la nueva versión.

11.8.3 Riesgos relacionados con el proceso

Las siguientes secciones se enfocan en describir posibles riesgos relacionados al proceso de generación de casos de prueba cuando se decide utilizar herramientas impulsadas por IA.

11.8.3.1 Aprobación de casos de prueba sin revisión adecuada

Existe el riesgo de que los ingenieros, especialmente aquellos bajo presión de tiempo, aprueben los casos de prueba generados por la herramienta sin llevar a cabo la revisión crítica requerida, confiando excesivamente en los resultados del modelo de IA.

Este riesgo puede anular completamente el valor de la revisión humana como mecanismo de control, convirtiendo efectivamente a la herramienta en un sistema de automatización completa de la verificación, lo cual no está permitido bajo el marco definido en esta guía ni es compatible con los principios del DO-178C.

El proceso de revisión debe estar formalmente definido y documentado, incluyendo una lista de verificación obligatoria que el ingeniero debe completar para cada caso de prueba antes de otorgar su aprobación. La organización debe establecer controles de proceso que hagan visible el esfuerzo de revisión y que desincentiven la aprobación superficial.

11.8.3.2 Uso de la herramienta fuera de su alcance definido

Los usuarios pueden intentar utilizar la herramienta para tipos de requerimientos que no están cubiertos por su alcance, como requerimientos basados en máquinas de estado, requerimientos con referencias a imágenes o diagramas, o requerimientos de sistemas para los cuales la herramienta no fue entrenada ni validada.

La calidad de los casos de prueba generados para requerimientos fuera del alcance es impredecible y no puede garantizarse. El uso de dichos casos de prueba en un proceso de verificación formal podría introducir una falsa sensación de seguridad.

La herramienta debe incluir mecanismos de detección que identifiquen cuando un requerimiento presenta características fuera de su alcance y debe notificar al usuario de manera explícita. El alcance de la herramienta debe estar documentado y ser conocido por todos los usuarios antes de su uso en proyectos reales.

11.8.4 Riesgos relacionados con el cumplimiento normativo

Las siguientes secciones se enfocan en describir posibles riesgos relacionados al cumplimiento de los estándares y guías cuando se decide utilizar herramientas impulsadas por IA.

11.8.4.1 Uso de la herramienta sin calificación bajo DO-330

Como se establece en el Principio Evitar la personificación de sistemas de inteligencia artificial., si la herramienta es utilizada de forma que reemplace o elimine una actividad de verificación que de otro modo requeriría revisión humana, debe ser calificada bajo DO-330. El riesgo consiste en que la herramienta sea utilizada de manera que exceda el nivel de automatización para el cual fue diseñada, sin que se haya llevado a cabo el proceso de calificación correspondiente.

El uso de una herramienta no calificada para automatizar actividades de verificación que requieren calificación puede resultar en la invalidación de las

evidencias de verificación generadas, comprometiendo el proceso de certificación del software aeronáutico.

El alcance funcional de la herramienta debe definirse explícitamente de manera que su uso no supere el nivel de asistencia al ingeniero. Cualquier expansión de capacidades que incremente el nivel de automatización debe ir acompañada de una evaluación del impacto sobre los requisitos de calificación bajo DO-330 antes de ser implementada.

11.8.4.2 Trazabilidad insuficiente entre casos de prueba y requerimientos

El DO-178C exige que exista trazabilidad bidireccional entre los requerimientos del software y los casos de prueba que los verifican. La herramienta puede generar casos de prueba sin vincularlos de forma explícita y documentada al requerimiento de origen y al criterio de evaluación que los motivó, dificultando la demostración de cobertura durante las auditorías de certificación.

La ausencia de trazabilidad adecuada puede resultar en observaciones o hallazgos durante las auditorías de la autoridad certificadora, requiriendo trabajo adicional de documentación y potencialmente retrasando el proceso de certificación.

El formato de casos de prueba definido en la Tabla 13 incluye los campos necesarios para garantizar la trazabilidad requerida, incluyendo el identificador del requerimiento verificado y el criterio de evaluación aplicado. El uso de este formato debe ser obligatorio para todos los casos de prueba generados y aprobados bajo esta guía.

11.8.5 Resumen de riesgos

La Tabla 14 presenta un resumen consolidado de los riesgos identificados, incluyendo su categoría, severidad estimada y el mecanismo de mitigación principal asociado.

Tabla 14: Resumen de riesgos identificados.

Descripción	Categoría	Severidad	Mitigación Principal
Casos de prueba faltantes	Calidad	Alta	Revisión contra Tabla 2

Descripción	Categoría	Severidad	Mitigación Principal
Resultado esperado incorrecto	Calidad	Alta	Aplicación obligatoria del criterio CI-8
Casos de prueba redundantes	Calidad	Baja	Revisión por ingeniero
Condiciones límite no evaluadas	Calidad	Alta	Aplicación rigurosa del criterio CI-2
Interpretación incorrecta del requerimiento	Modelo de IA	Alta	Validación previa de la interpretación
Variabilidad no determinística	Modelo de IA	Media	Control de configuración de casos aprobados
Degradación entre versiones	Modelo de IA	Media	Benchmarks por versión
Aprobación sin revisión adecuada	Proceso	Alta	Lista de verificación obligatoria
Uso fuera del alcance definido	Proceso	Media	Detección automática y notificación
Uso sin calificación DO-330	Normativo	Alta	Definición explícita del alcance funcional
Trazabilidad insuficiente	Normativo	Alta	Uso obligatorio del formato de la Tabla 13

12 Capítulo 12. Desarrollo de prototipo

Esta sección detalla cómo la implementación técnica del prototipo aborda y cumple directamente con los principios de seguridad aeroespacial, trazabilidad y determinismo definidos en el Capítulo 11. Guía de integración de modelos LLMs en procesos de identificación de casos de prueba.

12.1 Interfaz gráfica

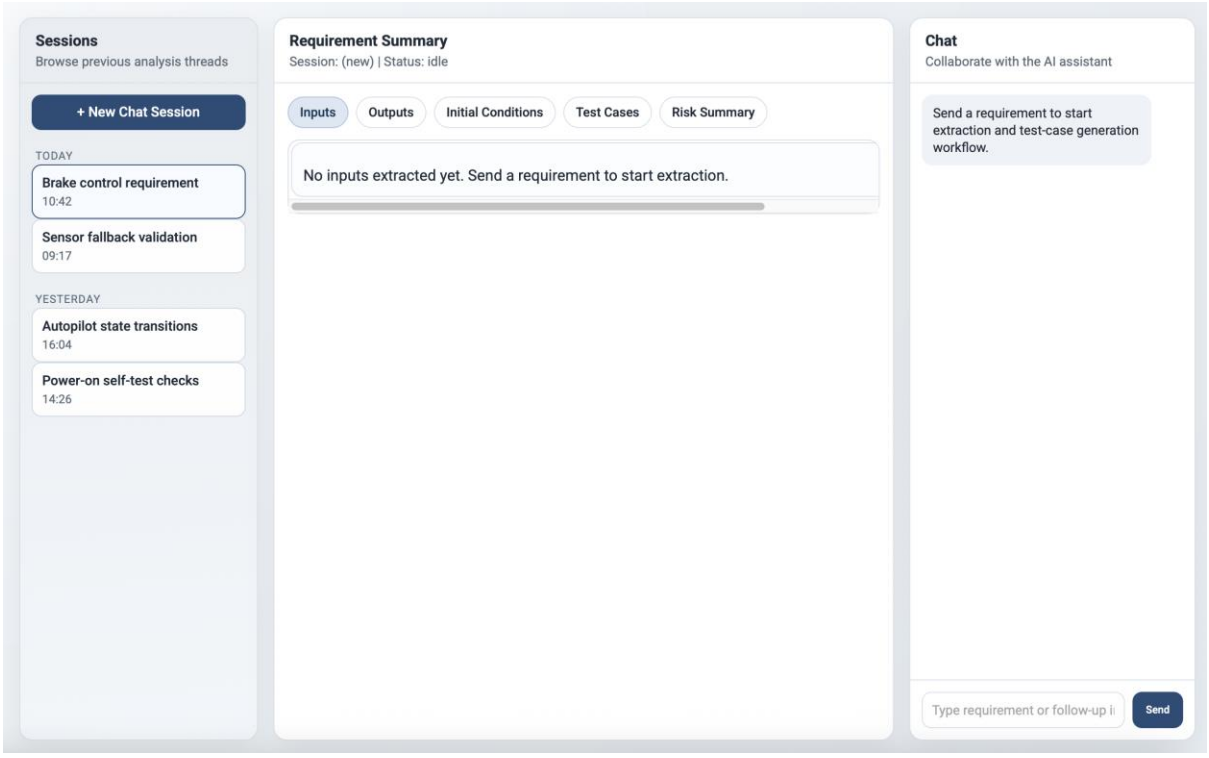
La Figura 23 muestra el diseño de la interfaz gráfica para el prototipo. Este diseño contiene tres columnas para mostrar información relevante para el usuario.

En la columna izquierda se muestran las sesiones previas del usuario que le permite mostrar los casos de prueba generados previamente. También, se incluye la opción de generar una nueva conversación.

En la columna del centro se incluye información relevante extraída del requerimiento, tal como variables de entrada, condiciones iniciales, variables de salida, riesgos identificados y casos de prueba.

La columna derecha contiene el chat en el que el usuario puede agregar la información del requerimiento a verificar y dar retroalimentación para modificar los datos generados por el modelo.

Figura 23: Interfaz gráfica del prototipo



Cabe destacar que el lenguaje de la aplicación es el inglés, ya que este estudio se enfoca en los lineamientos establecidos por la FAA del Departamento de Transporte de los Estados Unidos.

A continuación, se detalla el diseño del prototipo y su cumplimiento con las guías establecidas en el Capítulo 11. Guía de integración de modelos LLMs en procesos de identificación de casos de prueba.

12.2 Diseño del prototipo

Para diseñar el prototipo se consideraron varias aristas relacionadas con la generación de casos de prueba. Entre los principales temas que influyeron en el diseño del prototipo se encuentran: el manejo de riesgos al utilizar herramientas impulsadas por IA, el proceso de generación de casos de prueba y los lineamientos establecidos en la guía de integración de modelos de LLM.

12.2.1 Diseño de la arquitectura del software

El framework que se utilizó para desarrollar el flujo de trabajo de la aplicación fue LangGraph (LangChain, 2026). La Figura 24 muestra el flujo de trabajo diseñado para la generación de casos de prueba.

Por medio de LangGraph se diseñó un flujo de trabajo que combina encadenamiento de prompts, paralelización e interrupciones (LangChain, 2026).

El flujo comienza cuando el usuario pregunta generar los casos de prueba para un requerimiento dado. El primer nodo del flujo extrae la descripción del requerimiento incluyendo información relevante como tolerancias, resolución, tipo de requerimiento y rangos de operación de las variables.

Esta información es posteriormente procesada por la etapa de paralelización en donde existen tres flujos que se ejecutan en paralelo para realizar diferentes tareas que son independientes.

Un flujo analiza la información del requerimiento para identificar las condiciones iniciales de cada caso de prueba. Otro flujo se encarga de procesar la información obtenida para identificar las variables de salida y sus correspondientes posibles valores.

El tercer flujo tiene dos tareas principales. La primera consiste en identificar las variables de entrada del requerimiento y asignarles un tipo según el proceso descrito en las sección 6.2. Posteriormente, se identifican los valores para cada variable de entrada según el proceso descrito en la sección 6.3. Estos procesos también son compatibles con los lineamientos establecidos en la guía de integración de modelos (11.6).

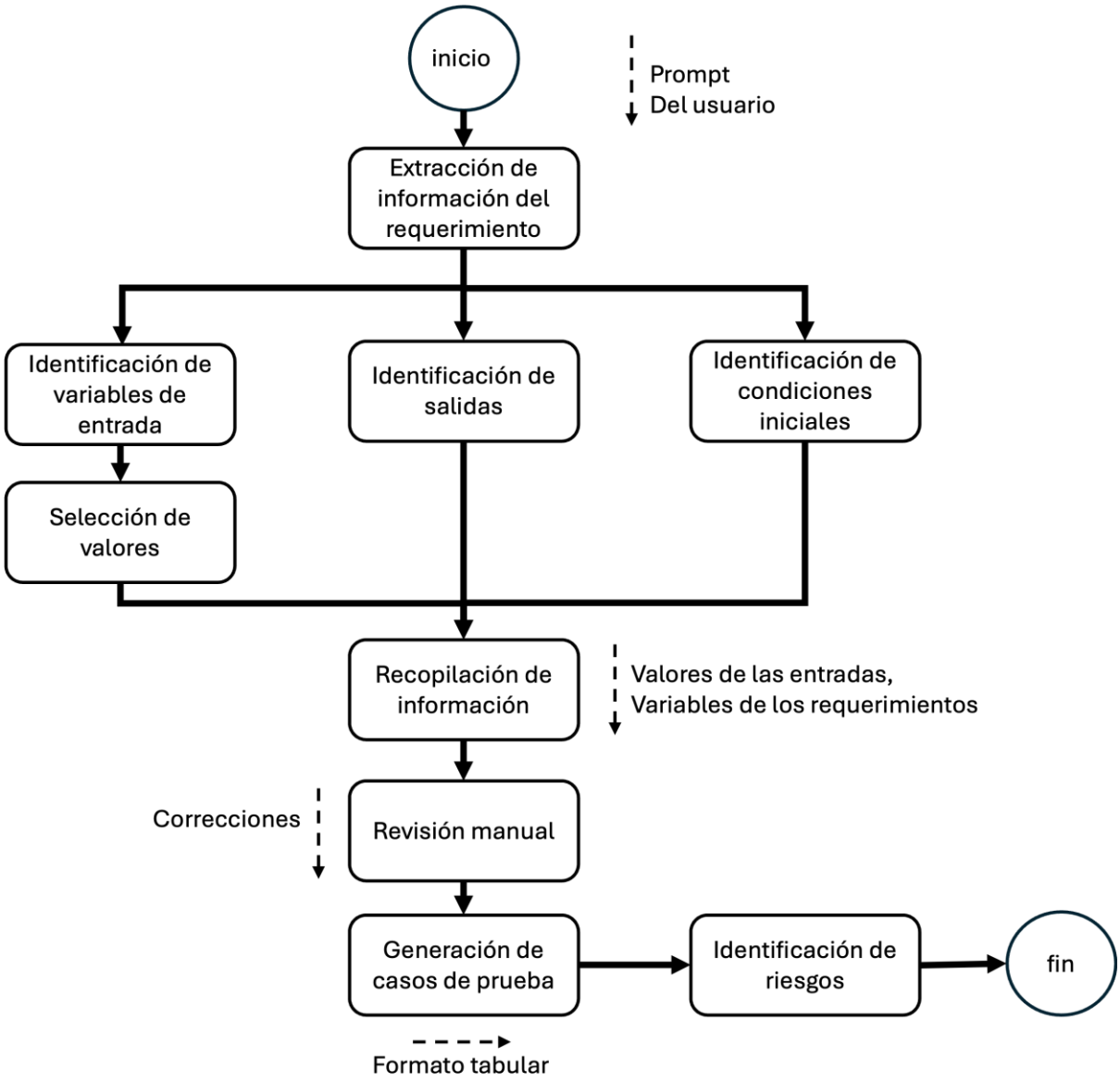
La información generada por cada flujo es recopilada por el siguiente nodo, responsable de mostrar las variables de entrada, salida y condiciones iniciales, junto con sus respectivos valores, en la interfaz gráfica.

Una vez generada la información en la interfaz gráfica, se realiza una interrupción en el flujo para esperar por retroalimentación del usuario. En el chat o en la interfaz gráfica el usuario puede realizar modificaciones a los valores generados por el modelo LLM y una vez se cuente con la aprobación del usuario, se procede a generar los casos de prueba.

El nodo encargado de generar los casos de prueba sigue el proceso establecido en la sección 11.7 en donde se utiliza la técnica de verificar una variable de entrada a la vez mientras a las otras variables se les asigna un valor fijo. Además se siguen los criterios establecidos para generar casos de prueba de acuerdo con el tipo de requerimiento y variable. Esta información se muestra en un formato tabular en la interfaz gráfica para el uso del ingeniero.

Finalmente, el nodo de riesgos analiza el requerimiento, los casos de prueba generados, el tipo de requerimiento generado para alertar al usuario de ciertos riesgos o medidas que se deben tomar para evitar y corregir errores en los resultados generados por la herramienta.

Figura 24: Flujo de trabajo del prototipo



12.2.2 Procesamiento de variables de requerimientos

Las variables de los requerimientos se muestran en la interfaz gráfica para que el usuario analice la información procesada por la aplicación y visualice de manera más clara los casos de prueba necesarios.

Las variables de entrada son presentadas al usuario con un formato diferente ya que el ingeniero de verificación debe analizar el tipo de variable entrada, rango

de operación, y otras variables para poder seleccionar todos los valores que son necesarios para verificar correctamente el requerimiento.

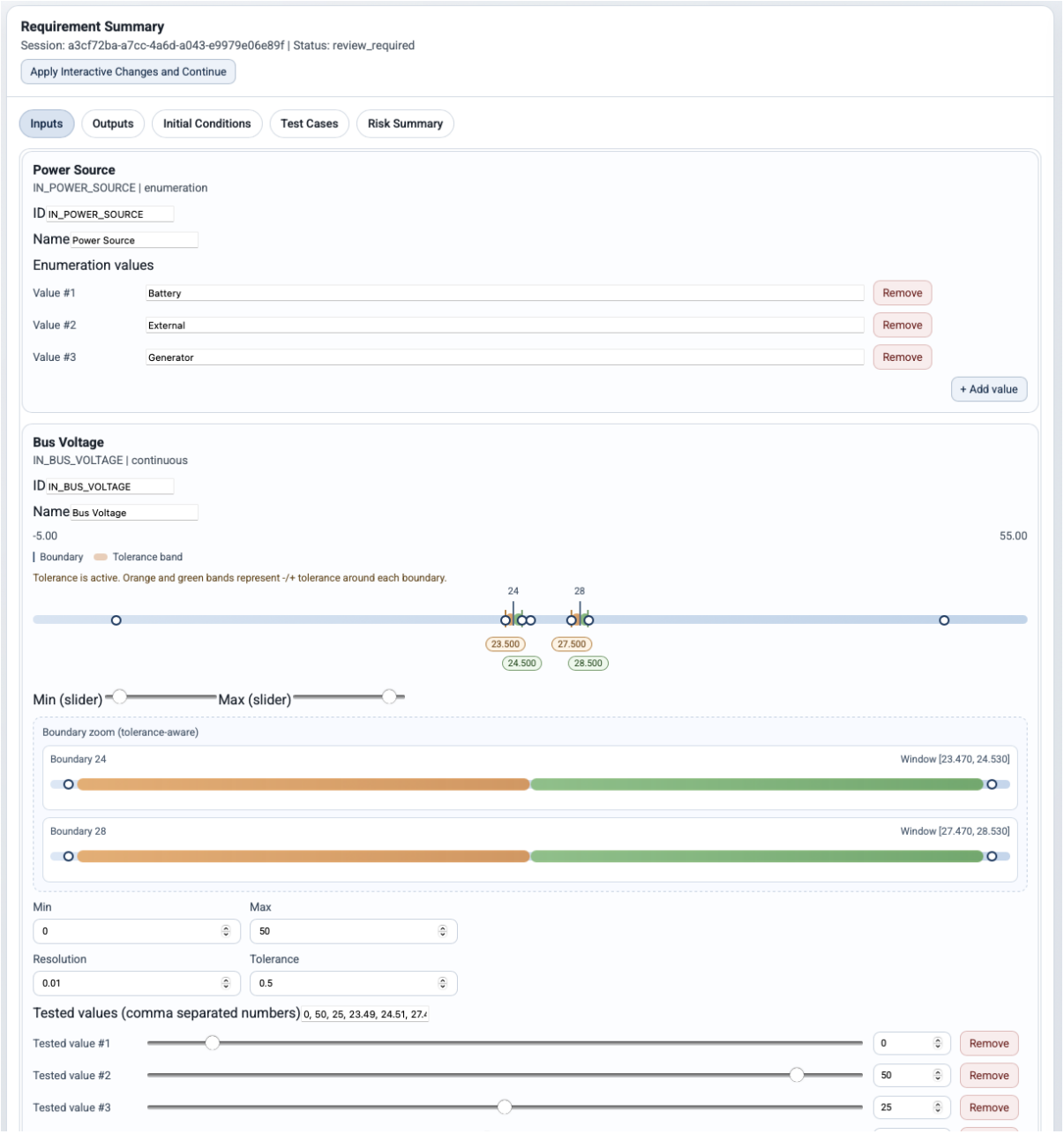
Por este motivo, el diseño de la pestaña de variables de entrada difiere del de las otras dos pestañas, las de variables de salida y condiciones iniciales. En este caso, dependiendo del tipo de variable de entrada se muestran los valores iniciales identificados por el modelo.

La Figura 25 muestra el diseño de la pestaña de las variables de entrada. Como se puede observar, el usuario puede modificar, agregar o eliminar los valores preseleccionados por el modelo.

En el caso de las variables de entrada clasificadas como de rango continuo, la visualización es diferente, ya que se presenta todo el rango al usuario y los valores identificados se muestran como puntos dentro del rango. El usuario puede modificar el rango, los valores seleccionados, la resolución y la tolerancia.

Para el caso de los límites identificados en el requerimiento, se ofrece otra visualización en donde se muestra un acercamiento del rango cerca de los límites para que el usuario pueda identificar de mejor manera los valores que son necesarios. Típicamente, las tolerancias son valores pequeños en comparación con el rango por lo que esto permite evidenciar los valores que son necesarios para verificar correctamente los límites de acuerdo con el proceso descrito en el Capítulo 8. Caso de estudio del proceso de identificación de casos de prueba.

Figura 25: Visualización de las variables de entrada

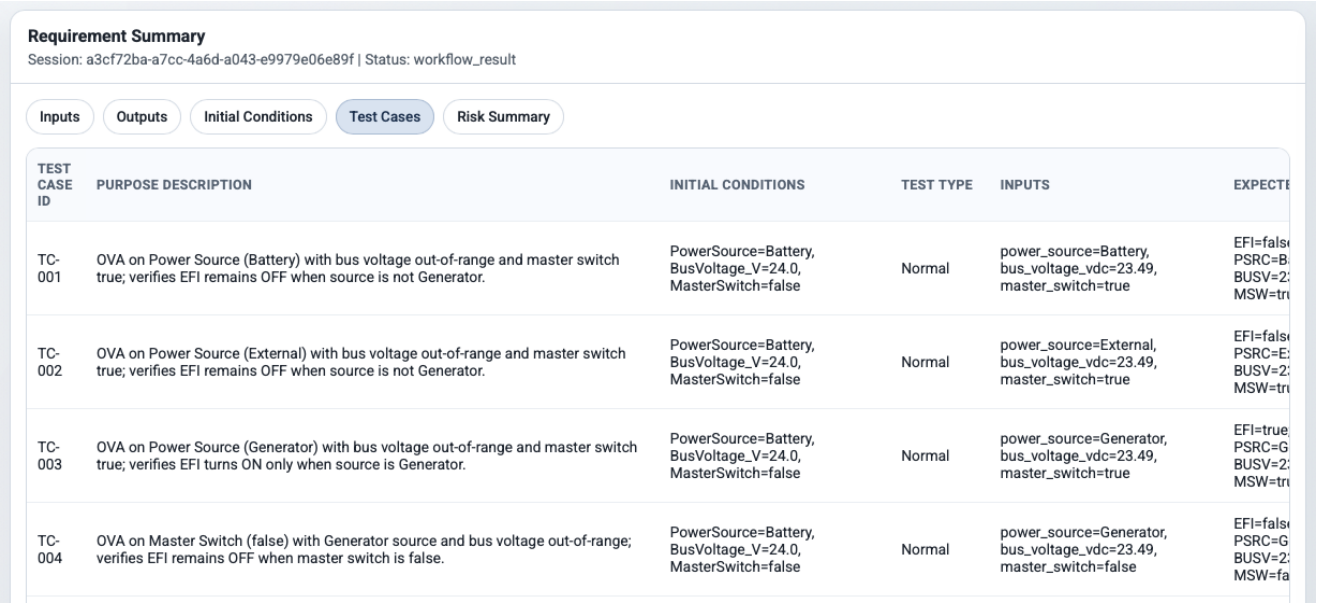


12.2.3 Casos de prueba

Después de recibir retroalimentación de parte del usuario, se procede a generar los casos de prueba. La Figura 26 muestra el formato en el que presentan los casos de prueba al usuario.

Cada fila corresponde a un caso de prueba generado. Cada caso de prueba contiene el identificador, descripción o propósito del caso de prueba, condiciones iniciales, tipo de prueba (normal o robusta), valores de entrada, valores esperados en cada prueba y el identificador de los requerimientos probados. El formato sigue los lineamientos sugeridos de la guía de integración para la Generación de casos de prueba.

Figura 26: Casos de prueba en el prototipo funcional

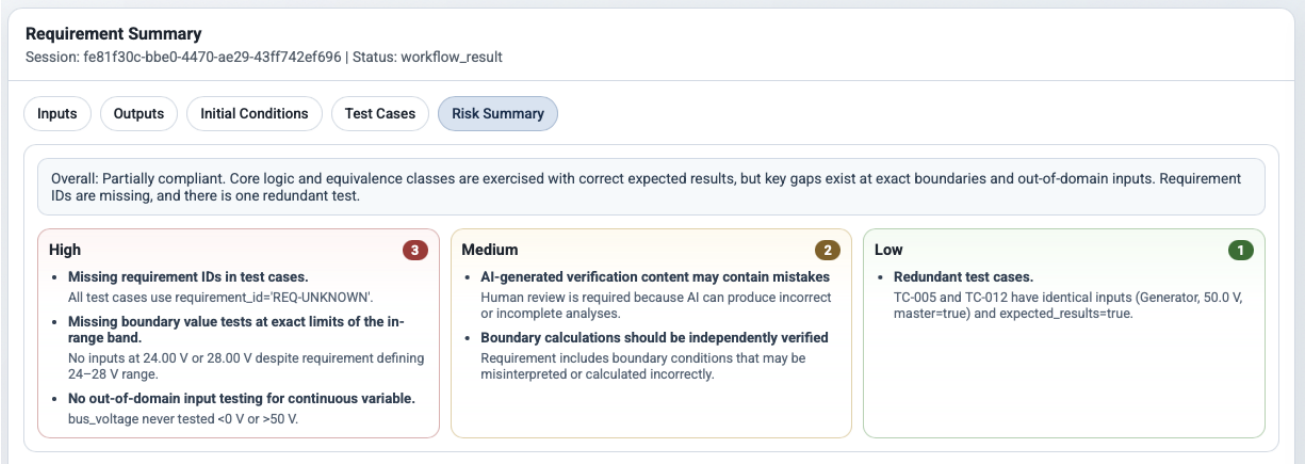


12.2.4 Manejo de riesgos

Se incluyó una funcionalidad para el control de riesgos con el fin de ayudar al ingeniero a detectar errores y también alertar al usuario de posibles errores asociados con el uso de esta aplicación.

La Figura 27 muestra la pestaña del control de riesgos. Se incluyen tres categorías para el control de riesgos dependiendo del nivel de impacto: alto, bajo y medio.

Figura 27: Control de riesgos dentro del prototipo



La muestra cómo se gestionan los riesgos en la aplicación. Estos riesgos coinciden con los riesgos identificados en la sección 11.8.5. Una vez generados los casos de prueba, el nodo responsable del control de riesgos ejecuta validaciones con base en el requerimiento y en los casos de prueba generados, para notificar al usuario sobre los posibles riesgos presentes que requieren acciones adicionales por parte del usuario.

Tabla 15: Implementación de riesgos dentro de la aplicación

Descripción	Control del riesgo dentro de la aplicación
Casos de prueba faltantes	LLM contiene lineamientos de aceptación de casos de prueba por lo que asiste en esta tarea. Revisión manual de los resultados por parte del usuario.
Resultado esperado incorrecto	LLM analiza los resultados esperados, los valores de entrada y la descripción del requerimiento para identificar posibles errores.
Casos de prueba redundantes	LLM analiza si hay casos de prueba duplicados o redundantes.
Condiciones límite no evaluadas	LLM analiza si los valores de entrada verifican completamente los valores límite. Se le indica al usuario que revise los valores sugeridos antes de generar los casos de prueba. Además, como estos casos involucran cálculos matemáticos se notifica al usuario que se deben revisar estos escenarios

Descripción	Control del riesgo dentro de la aplicación
	para asegurarse que los valores sugeridos son correctos.
Interpretación incorrecta del requerimiento	Intervención de parte del usuario para corregir cualquier error en la identificación de variables de salida, condiciones iniciales y salidas.
Variabilidad no determinística	Al usuario siempre se le indica que los resultados deben revisarse ya que los modelos de IA pueden cometer errores.
Uso fuera del alcance definido	Se notifica al usuario si se solicitó procesar requerimientos relacionados con imágenes o máquinas de estado.
Trazabilidad insuficiente	Se notifica al usuario si el identificador del requerimiento no ha sido especificado.

12.3 Alcance del prototipo

El prototipo genera casos de prueba a partir de la información brindada de un requerimiento. Este prototipo no cuenta con un servidor, bases de datos o página web. El prototipo no procesa documentos, múltiples requerimientos a la vez ni genera pruebas finales.

El propósito del prototipo es mostrar cómo se puede desarrollar una aplicación utilizando los lineamientos establecidos en la guía creada en este proyecto, para el uso correcto de herramientas impulsadas por IA para la generación de casos de prueba.

El prototipo no es un producto final y no se espera utilizarlo en un entorno real de producción.

13 Capítulo 13. Conclusiones y Recomendaciones

En esta sección se presentan las conclusiones y recomendaciones de este trabajo de investigación.

13.1 Conclusiones

Se identificaron y documentaron de forma empírica las limitaciones y desafíos críticos del uso de LLMs en tareas de generación de casos de prueba. El análisis cualitativo evidenció que la naturaleza probabilística de los modelos introduce vulnerabilidades como alucinaciones estadísticas, omisiones sistemáticas en el análisis de condiciones de frontera y una tendencia al sobretesteo (*over-testing*) que genera redundancia operativa . Asimismo, se determinó que modelos de menor escala como Mistral muestran una alta inestabilidad y fallas críticas al procesar lógica matemática y variables de rangos continuos, lo que resalta la urgencia de establecer controles cuando se utilizan herramientas impulsadas por IA para este tipo de tareas.

Se describió y fundamentó un proceso estándar de identificación de escenarios y casos de prueba recomendado por las directrices de la norma DO-178C, logrando establecer la base metodológica requerida para guiar la asistencia con IA . La incorporación de la lista de verificación basada en los criterios formales del estándar (como clases de equivalencia, análisis de límites, valores mínimos/máximos y variables de tiempo) proporcionó el marco de referencia objetivo necesario para evaluar las respuestas de los modelos y asegurar que las pruebas cumplan con las exigencias de las revisiones formales regulatorias.

Se estableció una estrategia sistemática mediante la clasificación de requerimientos funcionales en lenguaje natural bajo las categorías "con condiciones" y "sin condiciones", así como la tipificación de variables de entrada en booleanas, enumeradas y de rango continuo. Esta estructuración, apoyada en la sintaxis estandarizada EARS, facilitó la optimización de las estrategias de generación de pruebas, permitiendo guiar con precisión al LLM en la aplicación del método de análisis de "una variable a la vez" y aislando las complejidades lógicas para su correcta evaluación posterior.

Se realizó un análisis comparativo de los tres paradigmas de aplicación de LLM (RAG, *prompts* avanzados y agentes autónomos) utilizando los modelos GPT-5 y Mistral. Los resultados demostraron que el paradigma de *prompting* avanzado con GPT-5 logró la mayor precisión global, con un puntaje F1 promedio de 0.848 ; sin

embargo, el enfoque basado en agentes autónomos, estructurado en LangGraph, resultó ser el más aplicable para la complejidad aeroespacial, ya que garantizó una alta consistencia operativa entre iteraciones y maximizó la sensibilidad (*recall*) promedio a 0.895, reduciendo de forma significativa la omisión de casos de prueba esenciales. En contraste, la aplicación RAG mostró deficiencias al recuperar con precisión el contexto numérico y al calcular las tolerancias de frontera.

Se elaboró una guía de integración estructurada que cumple un propósito dual: regular el desarrollo de herramientas asistidas por IA y servir como material de referencia para el entrenamiento del personal técnico de verificación. Este marco metodológico se alineó rigurosamente con los siete principios fundamentales de la hoja de ruta de seguridad para la Inteligencia Artificial de la FAA (2024), definiendo mecanismos obligatorios de contención de errores, formatos tabulares para salvaguardar la trazabilidad bidireccional y un enfoque de despliegue controlado e incremental en la organización.

Se diseñó y desarrolló un prototipo interactivo y funcional que integra con éxito las normativas DO-178C, la guía de integración desarrollada en este proyecto, los aspectos éticos y la seguridad del software. El diseño del software mitigó activamente los riesgos del modelo al forzar la visualización interactiva de variables continuas, tolerancias, límites numéricos y análisis de control de riesgos directamente en su interfaz.

Se cumple el objetivo general de la investigación al evaluar el impacto de los modelos LLM en la aceleración de los procesos de verificación basados en requerimientos dentro de la industria aeroespacial. La investigación demostró de manera concluyente que, si bien estas herramientas digitales aportan un valor extraordinario al optimizar la eficiencia temporal en el desarrollo de propuestas iniciales de prueba, su naturaleza no determinística les impide cumplir con los requisitos de repetibilidad y consistencia exigidos para la calificación de herramientas automatizadas bajo la estricta norma DO-330. Por consiguiente, se concluye que las aplicaciones impulsadas por LLM en el ámbito aeronáutico no deben sustituir el análisis humano de forma autónoma; por el contrario, deben implementarse bajo el paradigma de "humano en el flujo de trabajo" (*human-in-the-loop*), operando estrictamente como un copiloto tecnológico donde el ingeniero de

verificación calificado retiene la total responsabilidad legal, la supervisión crítica y la aprobación final de la seguridad del software de aviónica.

13.2 Recomendaciones

Se recomienda analizar la viabilidad de utilizar LLMs para asistir en la verificación de casos de prueba. En este estudio, el análisis de los casos de prueba fue un proceso manual, lo que implicó largas horas de esfuerzo. Si los LLM pueden asistir en esta tarea con un porcentaje aceptable de precisión, el número de requerimientos que pueden probarse y el de iteraciones pueden incrementarse significativamente.

El proceso de generación de casos de prueba puede ampliarse para incluir otros tipos de requerimientos que no fueron considerados en este estudio, como requerimientos con imágenes, diagramas, máquinas de estado e incluso el procesamiento de varios requerimientos a la vez.

Existen varios enfoques para implementar flujos de trabajo agenticos que pueden explorarse en futuros trabajos. Por ejemplo, sistemas de multiagentes o agentes que planifican de manera más libre las acciones a implementar y utilizan herramientas disponibles según la tarea que deben realizar. Se recomienda explorar las ventajas y limitaciones de estos enfoques.

14 Capítulo 14. Reflexiones Finales

El desarrollo de esta investigación y los resultados obtenidos permiten reflexionar en profundidad sobre los cambios que la inteligencia artificial generativa está provocando en los procesos de ingeniería de verificación aeroespacial. La drástica reducción del esfuerzo temporal en la formulación de propuestas iniciales de escenarios de prueba evidencia el potencial para optimizar la productividad organizacional y mitigar el desgaste cognitivo del personal técnico. No obstante, las marcadas discrepancias analizadas entre modelos comerciales a gran escala y opciones de código abierto de escala moderada, sumadas a los errores cualitativos identificados como alucinaciones estadísticas, omisiones de fronteras numéricas y sobretesteo, operan como un recordatorio crítico sobre la naturaleza probabilística e

inherentemente falible de estas tecnologías. Estos hallazgos validan de forma contundente que los LLM no deben ser concebidos como sustitutos automatizados capaces de ser calificados de forma autónoma bajo la norma DO-330. Por el contrario, la rigurosidad, la perspectiva analítica y la responsabilidad legal de la seguridad aérea y del cumplimiento del estándar DO-178C permanecen y deben permanecer firmemente ancladas en el criterio del ingeniero humano, consolidando el diseño metodológico bajo el paradigma de "humano en el flujo de trabajo" (*human-in-the-loop*).

Por otra parte, una de las lecciones fundamentales de este proyecto de graduación radica en la necesidad imperativa de formular procesos, guías de integración y marcos de trabajo que sean estrictamente agnósticos respecto de un modelo o herramienta tecnológica específica. El ecosistema de la inteligencia artificial evoluciona a una velocidad exponencial sin precedentes, donde los modelos de lenguaje y sus paradigmas de aplicación pueden volverse obsoletos en cuestión de meses.

Al proponer metodologías estandarizadas y agnósticas firmemente alineadas con las directrices de seguridad de la FAA, se garantiza que las organizaciones aeroespaciales puedan integrar de manera transparente los nuevos avances de la industria y sustituir los núcleos de IA subyacentes a medida que surjan modelos más robustos, eficientes o con mejores capacidades de razonamiento numérico, salvaguardando así la resiliencia operativa, la soberanía de los datos propietarios y el cumplimiento normativo a largo plazo .

15 Capítulo 15. Trabajos a Futuro {Líneas Futuras de Investigación}

Los resultados obtenidos en esta investigación concluyeron que la asistencia de los LLM (modelos de lenguaje grande) tiene un impacto positivo al reducir el tiempo requerido para identificar el conjunto inicial de casos de prueba para un requisito determinado. Sin embargo, los hallazgos de esta evaluación también revelaron errores comunes introducidos al identificar dichos casos. Estos incluyeron:

- Casos de prueba faltantes: Ausencia de pruebas necesarias para proporcionar una muestra representativa de todo el rango de entrada.

- Errores matemáticos: Resultados incorrectos al realizar operaciones cuantitativas.
- Cálculos de valores límite inexactos: Fallas en la determinación de los valores frontera.
- Sobre-evaluación (over-testing): Inclusión de pruebas innecesarias que llevan a evaluar un requisito en exceso.

Al evaluar los casos de prueba generados directamente por los ingenieros, se observó que la falta o el error en los casos de prueba para verificar el rango completo de entrada y los límites eran algunos de los problemas más habituales. Las operaciones matemáticas y los cálculos de límites representan precisamente algunas de las actividades que requieren más tiempo al identificar las pruebas necesarias.

Considerando que estos también fueron parte de los inconvenientes comunes observados al generar casos de prueba con la asistencia de LLM, esta evaluación indica que los ingenieros tendrían que invertir tiempo adicional en corregir los casos de prueba incorrectos sugeridos por el modelo y en añadir los casos de prueba faltantes.

Si bien algunos paradigmas de aplicación complejos, como el flujo de trabajo agéntico, proporcionaron mayor estabilidad y consistencia en los resultados, aún se requieren mejoras adicionales para reducir los errores observados durante la investigación. Estas limitaciones no solo afectan la calidad general de los casos de prueba, sino que también reducen el ahorro potencial de tiempo para los ingenieros.

Las investigaciones futuras deberían enfocarse en:

- Explorar enfoques híbridos: Combinar los LLM con técnicas de programación tradicionales para mitigar los errores observados en esta evaluación. Los LLM pueden por ejemplo delegar tareas de cálculos matemáticos o selección de las combinaciones de casos de prueba a funciones determinísticas para reducir los errores y pueden enfocarse en el planeamiento de la estrategia a seguir para la generación de casos de prueba.

- Incorporar otros modelos de código abierto: Evaluar modelos con parámetros de mayor tamaño para examinar su rendimiento y la variabilidad de sus resultados. Los modelos abiertos representan una opción atractiva para las empresas que buscan un mayor control sobre la privacidad de sus datos.

Asimismo, este estudio se enfoca en procesar requerimientos individuales y no en grupo, por lo que se sugiere explorar este enfoque en futuros estudios. Usualmente, es necesario procesar múltiples requerimientos y e información de varios documentos a la vez para tener el panorama completo de la funcionalidad bajo prueba.

Por último, la generación de casos de prueba es uno de los primeros pasos para generar las pruebas necesarias para verificar un requerimiento según los lineamientos de la guía DO-178C. Se sugiere explorar otras actividades como la generación completa de las pruebas tomando en cuenta los casos de prueba identificados y el análisis de los requerimientos necesarios para incluir una estrategia de pruebas adecuada.

16 Referencias

Administration, F. A. (2024). *Roadmap for artificial intelligence safety assurance – Version I.* Federal Aviation Administration (FAA).

Alassan, M., Espejel, J., Bouhandi, M., Dahhane, W., & Ettifouri, E. (s.f.). Comparison of Open-Source and Proprietary LLMs for Machine Reading Comprehension: A Practical Analysis for Industrial Applications.

Alassan, M., López Espeje, J., Bouhandi, M., Dahhane, W., & Ettifouri, E. (2024). Benchmarking Open-Source Language Models for Efficient Question Answering in Industrial Applications.

Anthropic. (19 de December de 2024). *Building effective agents*. Obtenido de Anthropic : https://www.anthropic.com/engineering/building-effective-agents

Barnard, J. (2026). *What are embeddings?* Recuperado el Marzo de 2026, de IBM Think: https://www.ibm.com/think/topics/embedding

Brahma Reddy Korraprolu, P. P. (2025). Test Case Generation for Requirements in Natural Language - An LLM Comparison Study. *Software Engineering Conference* (págs. 1-5). New York: Association for Computing Machinery. doi:10.1145/3717383.3717389

Chroma. (2026). Obtenido de Chroma: https://www.trychroma.com

Coursaris, C., Beringer, J., Leger, P.-M., & Öz, B. (2025). *The design of human-centered artificial intelligence for the workplace.* Cham, Switzerland: Springer Nature Switzerland AG.

Geroimenko, V. (2024). *The essential guide to prompt engineering: Key principles, techniques, challenges, and security risks.* Cham: Springer Nature Switzerland.

Koon, S. (2024). Reasoning with AI: Six essential factors for augmenting human expertise in the workplace. En *he design of human-centered artificial intelligence for the workplace* (págs. 81–104). Springer Nature Switzerland.

Korraprolu, B., Pinninti, P., & Reddy, Y. (2025). Test case generation for requirements in natural language – An LLM comparison study. *Proceedings of the 18th Innovations in Software Engineering Conference (ISEC '25)* (págs. Article 9, 1-5). Association for Computing Machinery.

Li, K. Y. (2024). *Large language models as test case generators: Performance evaluation and enhancement.*

Li, X., Wang, S., Zeng, S., Wu, Y., & Yang, Y. (2024). A survey on LLM-based multi-agent systems: Workflow, infrastructure, and challenges. *Vicinagearth, 1*.

Malmberg, H. (2025). *Automated Test Generation From Software Requirements Using an LLM: Generating Black-Box Test Cases From Real-World Automotive Software Requirements With GPT-4o.* Disertación, KTH Royal Institute of Technology.

Mavin, A. (2019). *EARS (Easy Approach to Requirements Syntax)*. Obtenido de Alistair Mavin: https://alistairmavin.com/ears/

Oche, A. J. (2025). *A systematic review of key retrieval-augmented generation (RAG) systems: Progress, gaps, and future directions.*

RTCA(a). (2011). *DO-178C: Software considerations in airborne systems and equipment certification.* RTCA.

RTCA(b). (2011). *DO-330: Software Tool Qualification Considerations.* Washington, D.C.: RTCA.

Solutions, V. (28 de October de 2025). *¿Qué es DO-178C?* Obtenido de Visure Solutions: https://visuresolutions.com/aerospace-and-defense/do-178c/

Su, Q., Li, X., Ren, Y., Ouyang, X., Hu, C., & Yin, Y. (2024). State diagram extension and test case generation based on large language models for improving test engineers' efficiency in safety testing. *2024 IEEE 35th International Symposium on Software Reliability Engineering Workshops (ISSREW)* (págs. 287–294). IEEE.

Wang, J., Chang, Y., Wang, X., Wu, Y., Yang, L., Zhu, K., . . . Xie, X. (2024). A survey on evaluation of large language models. *ACM Transactions on Intelligent Systems and Technology, 15*(3), Article 39.

Yang, Z., Huang, R., Cui, C., Niu, N., & Towey, D. (2025). Requirements-based test generation: A comprehensive survey. *ACM Transactions on Software Engineering and Methodology*, Article 1.

17 Apéndices

17.1 Requerimientos utilizados para la validación y evaluación de los casos de prueba

Los requerimientos utilizados para el refinamiento de los prompts y paradigmas de aplicación pueden encontrarse en el siguiente enlace: https://drive.google.com/drive/u/0/folders/1S6BYedWyiUg7nHTcNe94djMPzi8OUnAa

17.2 Scripts utilizados para los paradigmas de aplicación

En el siguiente enlace se pueden encontrar los scripts utilizados para implementar los paradigmas de aplicación estudiados en esta investigación:

https://drive.google.com/drive/u/0/folders/1kdOutDvTiPaFmcJn_mdNn1XJ6AIwcT_H

17.3 Resultados obtenidos en la evaluación de los paradigmas de aplicación

En el siguiente enlace se pueden encontrar los resultados obtenidos durante esta investigación:

https://drive.google.com/drive/u/0/folders/1AbzAT0OgrBdfIBPhngTI8SbTY-lOPAtc

17.4 Repositorio del prototipo funcional

https://github.com/kmendez1393/PIA_02