Diseño de un Marco de Referencia Architecture Building Blocks(ABBs) para la Creación de Aplicaciones Desarrolladas Mediante Inteligencia Artificial GenIA

Autor
Ricardo Ramiro Rivera Betancourth

Director
Jorge Iván Romero Gelvez

Co-Director
Mauricio Garcés Restrepo

Universidad de Bogotá Jorge Tadeo Lozano
Facultad de Ciencias Naturales e Ingeniería
Especialización en Desarrollo de Bases de Datos

Bogotá - Colombia, Noviembre de 2025

Índice

Índice de figuras

Índice de tablas

Resumen

Actualmente en la mayoría de empresas modernas, los proyectos con impacto tecnológico afrontan enormes y diversos retos particularmente en la búsqueda de optimizar costos y tiempos de implementación, lo que a su vez abre una serie de nuevos desafíos como la escasez de talento técnico especializado, tiempos excesivos no estimados y retornos no favorables de la inversión. En este escenario común, la Inteligencia Artificial Generativa (GenIA) y arquitecturas con base en agentes surgen como una solución lógica y a la mano gracias a su globalización y facilidad de acceso, en particular el desarrollo de aplicaciones empresariales con Inteligencia Artificial Generativa (GenAI) y agentes autónomos representa una oportunidad transformadora para la industria, prometiendo incrementos de velocidad y reducciones de costos significativas. Sin embargo, esta oportunidad se convierte a la vez en una amenaza debido a vulnerabilidades críticas que puede ocasionar, código generado con defectos de seguridad, incumplimiento normativo (GDPR, SOC 2), alucinaciones de modelos que producen lógica de negocio errónea, y una ausencia de trazabilidad que pone en riesgo la integridad de sistemas empresariales.

Actualmente, no existe un framework único, estandarizado y operativo que guíe el desarrollo seguro, eficiente y escalable de aplicaciones en entornos GenAI/agentes, justamente por esa razón este trabajo de investigación aborda esta falencia mediante la propuesta de diseño de un framework de Architecture Building Blocks (ABBs), estructurado en 7 capas operativas más una capa transversal de seguridad, basado en estándares internacionales reconocidos (TOGAF 9.2, NIST AI RMF, ISO/IEC 42001, OWASP). El framework propone 42 componentes arquitecturales específicos (ABBs) agrupados en: (1) Gobernanza, Riesgo y Cumplimiento; (2) Ciclo de Vida de Desarrollo (3) Orquestación Cognitiva y Arquitectura con base en Agentes ; (4) Contexto Semántico Datos y Conocimiento; (5) Ingeniería de Plataforma y Herramientas; (6) Infraestructura y Modelos Fundacionales; y (7) Gestión del Talento y Gestión del Cambio, y la capa transversal de Seguridad que impacta todas las capas.

La metodología de investigación se fundamenta en la validación de la propuesta a través de múltiples fuentes como estándares del mercado, papers académicos, informes de empresas de investigación como McKinsey, Gartner, BCG y documentación técnica de líderes tecnológicos como AWS, Microsoft, Google, IBM.

El principal aporte que busca esta investigación es la de proponer el primer marco de referencia integral y agnóstico de la industria que gobierne el desarrollo asistido por IA, se espera que los resultados permitirán a las compañías y organizaciones acelerar su adopción de GenAI reduciendo riesgos, incrementando la velocidad de desarrollo y asegurando cumplimiento normativo

Es importante aclarar por supuesto que se trata de un trabajo inicial que busca abrir el espacio para nuevas investigaciones relacionadas, en las que se profundice algunos temas y actualice otros en la medida en que la tecnología avanza. De igual manera, el modelo propuesto intenta ser integral y le da una mirada amplia, pero su implementación deberá ser paulatina e incremental seleccionando los componentes que más le sea conveniente a la compañía de acuerdo a sus principios, madurez y estrategia.

En el texto se toma la libertad de utilizar anglisismos aceptados por la industria puesto que su traducción literal al español en muchos casos puede ser confusa.

Abstract

The adoption of generative AI (GenAI) and autonomous agents in software development promises faster delivery and significant cost reductions. However, critical vulnerabilities remain: security flaws in generated code, regulatory non-compliance (GDPR, SOC 2), model hallucinations, and lack of traceability. Currently, there is no standardized operational framework to guide the secure, efficient, and scalable development of GenAI/agent-based enterprise applications.

This research seeks to design a 7-layer architecture building block (ABB) framework based on TOGAF 9.2, NIST AI RMF, ISO/IEC 42001, and OWASP. It proposes 42 specific ABBs covering governance, AI-augmented SDLC, cognitive orchestration, semantic context (RAG), platform engineering, infrastructure/models, and talent management, as well as a cross-cutting security layer.

Keywords: Architecture Building Blocks, Generative AI, Autonomous Agents, NIST AI RMF, ISO/IEC 42001, TOGAF, RAG, AI Governance.

Glosario

ABB Architecture Building Block, bloque constructivo arquitectónico que define una capacidad lógica necesaria para la organización, independiente de tecnología específica.. 2

Agente Autónomo Sistema de IA capaz de planificar, ejecutar y corregir tareas de forma independiente con mínima supervisión humana.. 12

Alucinación Cuando un modelo de IA genera información inexistente o incorrecta con alta confianza.. 16

Framework En el contexto de Arquitectura Empresarial, un esquema lógico que estructura el pensamiento y la clasificación de componentes organizacionales.. 2

GenAI (Generative AI), Inteligencia Artificial Generativa capaz de crear nuevo contenido (texto, código, imágenes) basándose en patrones aprendidos.. 2

LLM Large Language Model, modelo de Lenguaje Masivo entrenado con grandes cantidades de texto para entender y generar lenguaje natural y código.. 7

Low-Code/No-Code Plataformas que permiten crear aplicaciones con mínima o ninguna programación tradicional, usando interfaces visuales.. 11

OWASP Open Web Application Security Project - Fundación sin fines de lucro que trabaja para mejorar la seguridad del software.. 7

Pro-Code Desarrollo de software usando lenguajes de programación tradicionales, orientado a profesionales técnicos.. 8

Prompt Instrucción o contexto que se proporciona a un modelo de IA para guiar su respuesta.. 21

RAG Retrieval-Augmented Generation, es la técnica que mejora las respuestas de IA recuperando información relevante antes de generar una respuesta.. 12

SBB Solution Building Block, bloque constructivo de solución que implementa uno o más ABBs usando tecnología específica.. 6

SDLC Software Development Life Cycle - Ciclo de Vida de Desarrollo de Software, proceso estructurado para crear aplicaciones.. 8

1. Introducción

En la actualidad, una parte importante de las decisiones que se toman en los comités de tecnología de empresas de todos los tamaños —desde pymes hasta corporaciones y big tech— se orienta a la definición de estrategias que permitan priorizar proyectos a partir de un análisis costo–beneficio capaz de persuadir al nivel ejecutivo (C–Level) para su aprobación. Más allá de la alineación estratégica, uno de los factores determinantes en esta evaluación es el costo de implementación tecnológica, condicionado principalmente por el tiempo de ejecución y la disponibilidad de talento especializado.

La presión por responder a necesidades empresariales urgentes exige obtener resultados en plazos cada vez más reducidos, ya que los objetivos de negocio difícilmente admiten demoras. No obstante, los retrasos en la ejecución no solo generan pérdidas económicas, sino también pérdida de tiempo, considerado uno de los recursos más valiosos en la gestión organizacional.

Otro aspecto crítico es la disponibilidad de personal técnico calificado. La escasez de talento en el mercado, sumada a la alta rotación en el sector tecnológico, aumenta el riesgo de no contar con los perfiles adecuados en el momento oportuno. En Colombia, por ejemplo, el Ministerio de Tecnologías de la Información y las Comunicaciones (MinTIC) reportó en 2022 un déficit cercano a los 80.000 desarrolladores de software, cifra que podría ascender a 112.000 para finales de 2025. [1]

En respuesta a este escenario, algunas organizaciones han comenzado a sustituir entornos de desarrollo integrados (IDE) y arquitecturas tradicionales por plataformas Low Code —que requieren poco o ningún código— y, más recientemente, por herramientas de generación automática de código basadas en Inteligencia Artificial generativa. Estas tecnologías prometen acortar de forma significativa los tiempos de desarrollo —pasando de meses o años a días o semanas— y reducir la dependencia de personal altamente especializado.

Sin embargo, la adopción de estas soluciones plantea riesgos relevantes, especialmente en lo relacionado con la integridad y seguridad de los datos corporativos. Este documento analiza el impacto potencial de estas tecnologías sobre las bases de datos empresariales, identifica riesgos y vulnerabilidades asociados, y propone los fundamentos para un framework de gestión de datos en entornos de generación de código con IA generativa.

2. Descripción del Problema

En el escenario actual de transformación digital acelerada, las organizaciones enfrentan la presión de implementar soluciones tecnológicas que reduzcan tiempos de desarrollo, optimicen recursos y respondan con agilidad a las demandas del mercado. En este contexto, han cobrado protagonismo dos tecnologías emergentes: las plataformas Low Code y la Inteligencia Artificial Generativa (GenAI).

El término GenAI

Las plataformas Low Code facilitan la construcción de aplicaciones mediante interfaces visuales y componentes predefinidos, lo que disminuye la necesidad de codificación manual y, en consecuencia, la dependencia de talento altamente especializado. A su vez, la GenAI aporta capacidades avanzadas de generación de texto, código, imágenes y análisis a partir de instrucciones en lenguaje natural, ampliando las posibilidades de automatización y eficiencia en los procesos de desarrollo.

El uso combinado o independiente de estas tecnologías promete ventajas significativas, tales como mayor velocidad de entrega, reducción de costos iniciales y democratización del desarrollo de software. No obstante, dichos beneficios se acompañan de riesgos relevantes que, de no ser gestionados de forma adecuada, pueden comprometer la seguridad , el cumplimiento normativo, la calidad técnica y la sostenibilidad de las soluciones.

Actualmente, no existe un framework único, estandarizado y ampliamente aceptado que oriente el desarrollo de aplicaciones empresariales en entornos Low Code y GenAI. Esta ausencia se traduce en múltiples desafíos para las organizaciones, entre los cuales destacan:

El término Framework

La improvisación de procesos de desarrollo sin lineamientos claros de seguridad, gobernanza y arquitectura.

La carencia de un catálogo de Architecture Building Blocks (ABBs) reutilizables que garanticen consistencia técnica y mitiguen riesgos.

El término ABB

La subestimación de amenazas específicas, como ataques a modelos de IA, fuga de datos sensibles o dependencia excesiva de un proveedor.

A este panorama se suma la falta de un marco regulatorio consolidado que oriente a las empresas en la adopción de estas tecnologías. Proyecciones de Gartner advierten que para el 2026, el 80 % de las organizaciones habrá incorporado algún tipo de desarrollo Low Code, lo cual incrementa la urgencia de establecer lineamientos sólidos de seguridad y gobernanza. Expertos como Jeff Williams, CTO de Contrast Security, sostienen que “las plataformas Low Code no son inherentemente más vulnerables que el código tradicional, pero los riesgos siguen siendo los mismos”. Asimismo, consultoras como Deloitte subrayan que la adopción de tecnologías emergentes exige replantear la gestión de riesgos, los controles y los procesos operativos para garantizar la continuidad del negocio.

En consecuencia, aunque el desarrollo empresarial basado en GenAI representa una oportunidad estratégica para acelerar la innovación y optimizar recursos, la ausencia de un modelo de referencia arquitectural —particularmente en la definición de ABBs— limita su adopción segura y expone a las organizaciones a riesgos significativos.

3. Objetivos

3.1. Objetivo General

Formular el diseño de un marco de referencia basado en Architecture Building Blocks (ABBs) que oriente el desarrollo de aplicaciones empresariales seguras, eficientes y escalables con generación de código mediante Inteligencia Artificial Generativa y arquitectura con base en agentes, asegurando la integridad de los datos, el cumplimiento normativo y la adecuada optimización de recursos.

3.2. Objetivos Específicos

- Analizar el estado actual de adopción de plataformas Gen AI en entornos empresariales, identificando sus principales beneficios, limitaciones y riesgos asociados.
- Identificar las vulnerabilidades y amenazas específicas que puedan comprometer la seguridad y la integridad de los datos en dichos entornos.
- Definir un catálogo de Architecture Building Blocks (ABBs) reutilizables que garanticen consistencia técnica, trazabilidad y escalabilidad en el desarrollo de aplicaciones.
- Establecer lineamientos de gobernanza, control y auditoría aplicables a Arquitecturas Híbridas con base en agentes y GenAI.
- Proponer un conjunto de métricas de evaluación que permita medir el impacto del framework en términos de eficiencia operativa, reducción de costos y mitigación de riesgos.

4. Requerimientos

4.1. Requerimientos de Negocio

- Optimización de recursos técnicos y humanos : El marco debe permitir orientar al negocio en la reducción de la dependencia de talento especializado y los costos de implementación tecnológica.
- Reducción de tiempos de desarrollo: Se busca ayudar a acelerar la entrega de aplicaciones empresariales mediante el uso adecuado de generación automática de código con IA.
- Gestión de riesgos tecnológicos emergentes: El marco de referencia debe identificar y guiar en la mitigación de amenazas y vulnerabilidades de seguridad

4.2. Requerimientos Funcionales

- Provisión de un catálogo de ABBs reutilizable: El marco debe ofrecer un catálogo estandarizado de Architecture Building Blocks (ABBs) reutilizables que promuevan la consistencia técnica, la trazabilidad y la escalabilidad en el desarrollo de aplicaciones.
- Definición de lineamientos de gobernanza, control y auditoría : El marco debe establecer pautas claras para la gobernanza, el control y la auditoría aplicables a arquitecturas híbridas con base en agentes y Gen AI.
- Identificación y gestión de vulnerabilidades y amenazas : El marco debe incluir procesos para analizar e identificar las vulnerabilidades y amenazas específicas que puedan comprometer la seguridad y la integridad de los datos en entornos GenAI.

4.3. Requerimientos de Implementación

- Integración con herramientas de IA Generativa: El marco debe ser compatible y operable con las herramientas de generación automática de código basadas en IA generativa más comunes en el mercado, permitiendo su integración fluida en los flujos de trabajo de desarrollo.
- Asignación de roles y responsabilidades en la adopción del framework : Se deben definir claramente los perfiles encargados de gobernanza, auditoría y mantenimiento del ABB.
- Documentación técnica y guía de implementación: El proyecto debe incluir manuales, artefactos y recursos visuales que faciliten su adopción en distintos contextos de arquitectura empresarial

5. Estado del Arte

A partir de la experiencia propia como co-lider de una empresa pyme dedicada al desarrollo de software, en la que definimos como estrategia de crecimiento el disminuir los costos de nuestros clientes a través del aprovechamiento de nuevas tecnologías de automatización sin código y mediante el uso de la inteligencia artificial, nos encontramos con un obstáculo relevante y es que nos dimos cuenta que en la actualidad no existe un proceso, marco de referencia, arquitectura o framework que nos señale el camino o por lo menos las buenas prácticas a tener en cuenta para llevar a cabo la implementación de este tipo de arquitecturas. Es así como parte de la motivación de este documento es precisamente generar un artefacto arquitectural que si bien no es una guía exacta de los pasos a seguir, si servirá como marco de referencia a empresarios, desarrolladores, profesionales de diferente formación, etcétera, para que logren adoptar estas nuevas tecnologías de la mejor forma posible mitigando en alguna medida los problemas inherentes a este tipo de soluciones tecnológicas Y es que la cada vez más creciente tendencia en la utilización de tecnologías de Inteligencia Artificial Generativa (GenAI) está redefiniendo el desarrollo de software a nivel global, este fenómeno ha permitido a las organizaciones reducir drásticamente los tiempos de entrega (time to market), ha permitido la diversidad de perfiles y conocimientos al momento de la creación de aplicaciones y optimizado recursos humanos y técnicos (Sothis, 2021; Gartner, 2023) [2] [3] . Sin embargo, se evidencia que esta aceleración tecnológica introduce nuevos riesgos: vulnerabilidades en el código generado automáticamente (Pearce et al., 2022) [4] , problemas de gobernanza de datos (ISMS Forum, 2024) [5] y dependencia de proveedores.

En este contexto, se corrobora la premisa inicial; no existe un marco estandarizado de Architecture Building Blocks (ABBs) adaptado a entornos Low Code y GenAI que integre seguridad, gobernanza, escalabilidad y eficiencia operativa. A pesar de lo anterior, existe mucha documentación sobre aspectos relacionados que deben tenerse en cuenta para avanzar en los objetivos trasados

Arquitectura empresarial y ABBs El estándar TOGAF® define los ABBs como componentes conceptuales reutilizables que guían la selección de Solution Building Blocks (SBBs) (The Open Group, 2018) [6] . Ledesma Alvear (2017) sistematiza frameworks como TOGAF, Zachman, DoDAF y FEAF [7], concluyendo que su efectividad depende de la adaptación al contexto tecnológico y organizacional, aportando fundamentos para integrar procesos de negocio en arquitecturas modulares, aplicables a ABBs en entornos Low Code.

El término SBB

Los marcos de referencia consolidados, como TOGAF (The Open Group, 2018) o Zachman, ofrecen bases sólidas, pero no contemplan de forma específica los retos emergentes de estas tecnologías.

Código generado con IA Estudios recientes han documentado vulnerabilidades significativas en código generado por IA. Pearce et al. (2022) encontraron que el 40 % del código contenía fallos de seguridad. Shah y Srikant (2023) [8] , en Generative AI for Developers, advierten sobre riesgos de licenciamiento, sesgos y calidad del código. Rivera Ladino (2025) [9] , en un análisis comparativo de LLMs, identificó que casi la mitad del código evaluado

presentaba vulnerabilidades críticas según OWASP

Los términos LLM y OWASP

Vulnerabilidades y ciberseguridad en IA Brundage et al. (2020) [10] , en The Malicious Use of Artificial Intelligence, catalogan amenazas intencionales y proponen estrategias de mitigación y sientan bases para ABBs que incluyan defensas contra ataques adversarios. El ISMS Forum (2024) [5] publica guías técnicas con OWASP útiles para ABBs de seguridad y cumplimiento.

Plataformas y arquitecturas Low Code Sothis (2021) [2] describe beneficios y retos de plataformas como OutSystems y Mendix, mientras que el Magic Quadrant de Gartner (2023) [3] analiza capacidades clave y tendencias del mercado en relación al avance significativo de su uso, presenta casos de uso de Low Code con IA, destacando la importancia de DevSecOps y cumplimiento normativo.

El análisis anterior evidencia que si bien existen marcos consolidados de arquitectura empresarial como TOGAF, Zachman o FEAF, y abundante literatura sobre plataformas Low Code y código generado por IA, ninguno de estos enfoques aborda de manera integral los retos combinados que surgen cuando ambas tecnologías convergen en entornos corporativos.

Este trabajo, por tanto, no solo se apoya en fundamentos teóricos y prácticos consolidados, sino que también propone una innovación metodológica y que puede ser aplicada: la creación de un marco que pueda ser adoptado por organizaciones para maximizar los beneficios de GenAI, minimizando sus riesgos y asegurando la sostenibilidad tecnológica en el tiempo.

6. Marco Teórico

6.1. Introducción: Un cambio trascendental en la Ingeniería de Software

La industria de la tecnología está en un momento clave de cambio, mientras que existe mucha demanda de aplicaciones digitales (especialmente en el contexto de la transformación digital) aún no se ha satisfecho con un número adecuado de programadores calificados. Esto ha impulsado el uso de herramientas de Inteligencia Artificial Generativa (GenAI) en el proceso de desarrollo de software o Ciclo de Vida del Desarrollo de Software (SDLC). Estas herramientas ya no son solo una ayuda, están cambiando la forma de hacer el trabajo de programación.

El término SDLC

Visto desde la perspectiva de la Arquitectura Empresarial (EA) a un nivel general, este documento proporcionará un marco teórico. De esta manera, busca construir Bloques de Construcción de Arquitectura (ABBs). Estos ABBs sirven como guía para tecnologías "Pro-Code" asistidas por IA, como GitHub Copilot, Cursor o Google Jules. No incluimos plataformas "Low-Code", aquellas que utilizan interfaces visuales que son solo básicas, ya que nuestro énfasis estará en la programación profesional avanzada.

El término Pro-Code

Este marco es crucial, no solo para las operaciones diarias, sino también para la estrategia de la empresa. Tener una base teórica es, como muestran los estudios de Hernández Sampieri [11] y Méndez, clave para estudiar e incorporar nuevos conceptos. En el mundo actual, las empresas son desafiadas por sus líderes a seleccionar proyectos con rendimiento comprobado a menores costos. Pero el mercado laboral es un problema: datos del Ministerio de Tecnologías de la Información y las Comunicaciones (MinTIC) de Colombia sugieren que para 2025, podría haber una escasez de hasta 112,000 desarrolladores. [12] Esta escasez de talento, además de una alta rotación de empleados, significa que "el tiempo" es la moneda.

GenAI promete reducir ese tiempo de desarrollo de meses a semanas. Sin embargo, esto lleva a un problema: los humanos no pueden revisar todo el código generado tan rápidamente, lo que puede introducir errores en calidad o seguridad. En los informes de OWASP [13] [14] se advierte sobre tales riesgos potenciales como la vulnerabilidad del código y problemas sobre la propiedad intelectual si no se gestionan adecuadamente.

6.1.1. La Solución: Un Marco de Gobernanza

Este documento propone una solución formal basada en la Arquitectura Empresarial (EA), un conjunto de bloques arquitectónicos conocidos como Architecture Building Block (ABBs) que actúan de manera similar a "piezas de LEGO" o bloques estándar. Estos bloques fomentarán que las organizaciones adopten de manera segura la inteligencia artificial generativa en el ambiente tecnológico, definido a través de marcos de referencia arquitectónicos mediante los cuales podemos saber cómo organizar, gobernar y optimizar adecuadamente el desarrollo de software mientras trabajamos con herramientas de inteligencia artificial generativa. El marco se fundamenta en TOGAF (The Open Group Architecture Framework) [15] , que es uno de los marcos de arquitectura empresarial más populares y utilizados a nivel mundial, no se evalúa herramientas particulares (Solution Building Block o SBBs (que cambian rápidamente)) sino

que proporciona capacidades lógicas (ABBs) que existen a largo plazo.

6.2. Fundamentos: Entendiendo la Arquitectura Empresarial

6.2.1. ¿Qué es Realmente un "FrameworK"?

En primera medida se aclara un concepto puede generar alguna confusión. En el mundo tecnológico, la palabra "framework" se usa para referirse a muchas cosas diferentes, en este documento concretamente nos referimos a framework arquitectónico, no de una librería de código o una herramienta de software.

Para ser más explícito como analogía se puede pensar en un framework arquitectónico como los planos estandarizados de una casa:

- Los planos muestran dónde deben ir las habitaciones, las instalaciones eléctricas y de plomería.
- No importa qué marca de cables o tuberías compres; los planos siguen siendo válidos.
- Diferentes constructores pueden usar estos mismos planos para construir casas similares pero adaptadas a sus necesidades.

De manera similar, un marco de referencia define qué capacidades debe tener una organización para adoptar IA generativa de forma segura, sin prescribir qué productos específicos debe comprar.

6.2.2. ¿Qué es TOGAF y por qué es importante?

TOGAF es un marco de trabajo que proporciona un método estructurado y probado para diseñar, planificar, implementar y gobernar arquitecturas empresariales. TOGAF sería entonces como un manual de mejores prácticas utilizado por organizaciones líderes en todo el mundo para asegurar que sus sistemas tecnológicos estén bien organizados, sean escalables y alineados con los objetivos del negocio. [16]

La Arquitectura Empresarial (EA, por sus siglas en inglés) es la disciplina que organiza y estructura todos los componentes de una organización: sus procesos de negocio, sistemas de información, datos y tecnologías. Es como el plano maestro de un edificio complejo que muestra cómo todas las partes se conectan entre sí.

6.2.3. Architecture Building Blocks (ABBs): Los Componentes Fundamentales

Según el estándar TOGAF (The Open Group Architecture Framework), existen dos tipos de bloques arquitecturales:

1. Architecture Building Blocks (ABBs) el "Qué": Un ABB es una capacidad lógica que la compañía u organización necesita tener definida en términos de requisitos de negocio más no en términos de tecnología específica. Sus características principales son:

- Duraderos: No cambian con las tendencias tecnológicas
- Agnósticos de proveedor: No mencionan marcas o productos específicos
- Estratégicos: Se alinean con los objetivos de largo plazo de la organización.

2. Solution Building Blocks (SBBs), el "Cómo": Un SBB es la implementación física de uno o más ABBs usando tecnología específica. Sus características principales son:

- Transitorios: Pueden cambiar cada año o incluso cada trimestre
- Específicos de proveedor: Mencionan productos y marcas concretas
- Tácticos: Resuelven problemas operativos inmediatos

Una analogía práctica sería la siguiente; si se está diseñando una aplicación de comercio electrónico, un ABB sería "Sistema de Autenticación de Usuarios", este ABB define qué debe hacer (verificar identidades, gestionar sesiones, proteger credenciales), pero no especifica qué se usará OAuth, JWT, o autenticación biométrica. Esa implementación específica vendría después, en lo que TOGAF llama Solution Building Blocks (SBBs).

Tabla Comparativa		
Aspecto	**ABB (Architecture)**	**SBB(Solution)**
Naturaleza	Conceptual, abstracta	Concreta, física
Función	Define el problema y requisitos	Provee la solución técnica
Vida útil	Largo plazo (años)	Corto/medio plazo (meses)
Ejemplo en IA	"Mecanismo de inferencia de código seguro"	.ºpenAI Codex v2 vía API Azure"
En gobernanza	Estándar para auditorías	Objeto auditado

Tabla 1: Tabla Comparativa ABB vs SBB

6.2.4. El Ciclo de Vida del Desarrollo de Software (SDLC)

El Software Development Life Cycle (SDLC) es el proceso estructurado que siguen los equipos de desarrollo desde que conciben una idea hasta que el software está operando y se mantiene en el tiempo.

Las fases tradicionales del SDLC son:

- Planificación: Definir el alcance, objetivos y viabilidad del proyecto.
- Análisis de Requisitos: Documentar qué necesita el sistema hacer.
- Diseño: Crear la arquitectura y especificaciones técnicas
- Desarrollo/Codificación: Escribir el código del sistema.

- Pruebas: Validar que el software funciona correctamente
- Implementación: Desplegar el sistema en producción.
- Mantenimiento: Dar soporte continuo y realizar mejoras.

¿Cómo cambia esto con GenAI?

La IA Generativa está transformando cada una de estas fases:

- En Análisis: puede generar documentación técnica y casos de uso.
- En Diseño: puede sugerir arquitecturas basadas en patrones probados.
- En Desarrollo: puede generar código completo a partir de especificaciones.
- En Pruebas, puede crear casos de prueba automáticamente y detectar anomalías.

6.3. Pilares Conceptuales del Marco de Referencia

Este marco se sustenta en tres pilares fundamentales que se debe comprender y asimilar para garantizar su utilidad al máximo en el desarrollo asistido por IA.

6.3.1. Pilar 1: Paradigmas de Desarrollo de Software

Existen tres enfoques principales para desarrollar aplicaciones, cada uno con diferentes niveles de control y flexibilidad:

Paradigma	¿Qué es?	Ventajas	Desventajas	Ejemplo
No-Code	Plataformas 100 % visuales sin escribir código	Desarrollo extremadamente rápido; no requiere conocimientos de programación	Muy limitado en personalización; difícil de escalar	Crear un sitio web en Wix o una app simple en Bubble
Low-Code	Interfaces visuales + código personalizado cuando se necesita	Balance entre rapidez y flexibilidad; desarrollo hasta 10x más rápido	Menos control que pro-code; dependencia del proveedor de la plataforma	Power Apps de Microsoft, OutSystems
Pro-Code	Desarrollo tradicional escribiendo código completo	Máxima flexibilidad, control total, escalabilidad ilimitada	Requiere programadores expertos; desarrollo más lento	Aplicación empresarial en Java, Python, o React

Tabla 2: Principales enfoques para desarrollar aplicaciones

Los términos Low-Code/No-Code

Un ejemplo para clarificar estos conceptos sería el siguiente:

- **Low-Code:** Usa un editor visual de sitios web (como Wix o Squarespace)
- **Pro-Code:** Escribe HTML, CSS y JavaScript con sugerencias inteligentes de un asistente.

Este marco se enfoca en el desarrollo Pro-Code porque mantiene el control total y la flexibilidad, pero asistido por IA Generativa para acelerar el proceso buscando en la medida de lo posible no sacrifica la calidad, ni la seguridad.

- Desarrollo tradicional con lenguajes de programación (Python, Java, C++, etc.)
- Orientado a ingenieros de software profesionales
- La IA colabora en la creación del código, no lo oculta
- Mantiene la flexibilidad y potencia del código tradicional.

6.3.2. Pilar 2: Inteligencia Artificial Generativa en el Desarrollo de Software

Evolución: La tecnología detrás de estas herramientas ha evolucionado rápidamente [17]:

Fase 1: Modelos Generalistas (2020-2021):

- Ejemplo: GPT-3
- Capacidad: Podía generar código básico, pero era inconsistente
- Limitación: No entendía profundamente la estructura del código

Fase 2: Modelos Especializados (2021-2022)

- Ejemplo: OpenAI Codex (base de GitHub, Copilot)
- Capecidad : Entrenado específicamente con miles de millones de líneas de código de GitHub
- Mejora: Comprende sintaxis y semántica de múltiples lenguajes

Fase 3: Asistentes Contextuales (2022-2023)

- Ejemplo: Cursor IDE
- Tecnología: RAG (Retrieval-Augmented Generation)
- Capacidad: No solo mira el archivo abierto, sino que "entiende" todo el proyecto
- Cómo funciona: Indexa vectorialmente tu código completo para dar sugerencias más precisas

El término RAG

Fase 4: Agentes Autónomos (2024-presente)

- Ejemplo: Google Jules
- Capacidad: Puede planificar tareas, ejecutar comandos, crear pruebas, corregir errores
- Cambio paradigmático: Ya no solo sugiere, sino que actúa de forma más autónoma

El término Agente Autónomo

¿Qué es GenAI? La Inteligencia Artificial Generativa se refiere a sistemas de IA que pueden crear contenido nuevo (código, texto, imágenes) basándose en patrones aprendidos de enormes cantidades de datos. En el contexto del desarrollo de software, GenAI utiliza Modelos de Lenguaje Grandes (LLMs) entrenados con millones de líneas de código para generar, completar y optimizar código automáticamente.

Algunos casos de uso principales:

- Generación de código: Crear funciones, clases o módulos completos desde descripciones en lenguaje natural
- Refactorización: Optimizar código existente para mejorar rendimiento y legibilidad
- Documentación: Generar automáticamente comentarios y documentación técnica
- Pruebas: Crear casos de prueba unitarios y de integración
- Depuración: Identificar y sugerir correcciones para errores
- Revisión de código: Detectar vulnerabilidades de seguridad y problemas de calidad

RAG (Retrieval-Augmented Generation) RAG es una técnica avanzada que mejora la precisión de la IA Generativa. En lugar de depender únicamente de lo que el modelo aprendió durante su entrenamiento, RAG busca información actualizada y específica de bases de datos, documentación técnica o repositorios de código antes de generar una respuesta. [18]

Sin RAG, un LLM podría generar código basado en bibliotecas obsoletas o información incorrecta ("alucinaciones"), en su lagar RAG facilita que el sistema primero consulta la documentación oficial más reciente de un proyecto, asegurando respuestas precisas y actualizadas. [19]

Como caso práctico se solicita a una IA "genera código para conectarse a nuestra base de datos", primero buscará en tu documentació interna las credenciales,quemas y estándares específicos de la empresa antes de generar el código en lugar de crear un ejemplo genérico.

Agentes Autónomos de Codificación Los agentes autónomos son sistemas de IA que pueden planificar, ejecutar y completar tareas de desarrollo de software con mínima intervención humana. A diferencia de asistentes de IA simples que solo sugieren código, los agentes autónomos pueden:

- Entender un objetivo de alto nivel (eje: "implementa un sistema de notificaciones por email")
- Descomponerlo en subtareas (diseñar la arquitectura, escribir el código, crear pruebas)
- Ejecutar cada tarea secuencialmente.
- Autocorregirse cuando detectan errores

Diferencia clave: Los asistentes de IA requieren guía constante del desarrollador, generan fragmentos de código, y los agentes autónomos pueden manejar flujos completos del SDLC con intervención mínima.

Impacto en el Ciclo de Vida del Desarrollo (SDLC) En la práctica, hablando en el contexto de desarrollo de software, la IA generativa está impactando significativamente en cada fase del ciclo de vida que tradicionalmente conocíamos, así:

- Diseño: Asistencia en la definición de arquitecturas de microservicios, Sugerencia de patrones de diseño apropiados
- Codificación: Reducción drástica del "código repetitivo"
- Testing: Generación automática de pruebas unitarias, de integración y de estrés
- Documentación: Creación y actualización automática de documentación técnica, Sincronización permanente entre código y documentación
- Depuración: Análisis de causa raíz de errores, Sugerencia de correcciones ("fix capability")

En este sentido, varios estudios confirman efectivamente el aumento de productividad que puede ayudar a mitigar la falta de talento disponible (MinTIC 2024) [12], pero introduce desafíos en calidad y seguridad.

6.3.3. Pilar 3: Gobernanza de IA en el Desarrollo de Software

La integración de modelo estocásticos (probabilísticos) en procesos de ingeniería que requieren determinismo (exactitud) introduce una nueva categoría de riesgos empresariales. El análisis del "Estado del Arte" revela una preocupación creciente en la comunidad académica y profesional sobre las implicaciones de seguridad de la GenAI, una estrategia correcta de abordar estos inconvenientes se logra con un robusto marco de gobernanza sobre el ecosistema tecnológico emergente de la IA.

¿Qué es la gobernanza de IA? La gobernanza de IA es el conjunto de políticas, procesos y herramientas que las organizaciones usan para gestionar sistemas de IA de manera segura, ética y conforme a regulaciones. En el contexto del desarrollo asistido por IA, significa establecer reglas claras sobre:

- Qué código generado por IA puede usarse en producción
- Cómo validar la calidad y seguridad del código generado
- Cómo rastrear y documentar decisiones arquitecturales hechas con asistencia de IA
- Cómo proteger la propiedad intelectual y datos sensibles

Componentes principales de un marco de gobernanza:

1. Políticas y estándares: Reglas claras sobre el uso aceptable de herramientas GenAI
2. Evaluación de riesgos: Identificar posibles problemas (sesgos, vulnerabilidades, errores)
3. Auditoría y trazabilidad: Mantener registros de qué código fue generado por IA y cómo
4. Validación continua: Pruebas automatizadas para verificar calidad y seguridad
5. Responsabilidad humana: Los desarrolladores siguen siendo responsables del código final

Este documento se apoya principalmente en frameworks reconocidos que guían la implementación responsable de IA:

- NIST AI Risk Management Framework (RMF) [20]: Marco del Instituto Nacional de Estándares y Tecnología de EE.UU. para identificar y gestionar riesgos de IA
- ISO 42001: Estándar internacional para sistemas de gestión de IA [21]
- Principios de la OCDE sobre IA: Guías éticas para IA confiable desarrolladas por la Organización para la Cooperación y el Desarrollo Económicos

Evidencia de Vulnerabilidades Un estudio realizado sobre este tema "Asleep at the Keyboard?" (Pearce et al., 2022) [22], proporciona evidencia contundente sobre la "permisividad" de los desarrolladores que utilizan asistentes de IA tienen una propensión significativa a aceptar código con vulnerabilidades de seguridad a menudo con una falsa confianza en la calidad de la sugerencia de la máquina. Pearce encontró que hasta el 40 % del código generado en ciertos contextos contenía fallos explotables.

En esta misma línea, investigaciones más recientes como la de Rivera Ladino (2025) han realizado análisis comparativos de vulnerabilidades en código generado por diversos Modelos de Lenguaje (LLMs) [23]. Los hallazgos son críticos para nuestra arquitectura, casi la mitad del código evaluado presentaba vulnerabilidades críticas clasificadas por OWASP, esto confirma que el riesgo no es teórico ni aislado, sino sistémico e inherente a los datos de entrenamiento de los modelos actuales, los cuales a menudo aprenden de código legado inseguro presente en repositorios públicos .

Riesgos OWASP para LLMs La Fundación OWASP ha estandarizado estos riesgos en su "Top 10 para Aplicaciones LLM"[24], proporcionando un vocabulario común para la gestión de riesgos que nuestro framework debe adoptar:

- LLM02 - Manejo Inseguro de Salidas: Ocurre cuando el código generado por la IA se ejecuta o integra sin validación, permitiendo inyecciones (XSS, SQLi) indirectas.

- LLM03 - Envenenamiento de Datos de Entrenamiento y Alucinaciones: Incluye la "alucinación de paquete", donde la IA sugiere importar una librería que no existe. Atacantes humanos han comenzado a publicar paquetes maliciosos con los nombres que las IAs suelen alucinar, creando ataques a la cadena de suministro de software automatizados.

  El término Alucinación

- LLM06 - Divulgación de Información Sensible: Quizás el riesgo más agudo para la empresa. Sucede cuando los desarrolladores envían fragmentos de códig propietario, claves de API o datos de clientes a un modelo público (como ChatGPT o versiones no empresariales de Copilot) para su análisis, perdiendo el control sobre la confidencialidad de la información . [25]

Cumplimiento Normativo y Ético Más allá de la seguridad técnica, existen riesgos legales y éticos. Shah y Srikant (2023), en su obra "Generative AI for Developers", [17]advierten sobre los riesgos de licenciamiento: un modelo entrenado con código GPL podría generar sugerencias que, al integrarse en software propietario, "contaminen" legalmente la base de código de la empresa, obligándola a liberar su propiedad intelectual. Asimismo, Brundage et al. (2020) discuten el uso malicioso de la IA y la necesidad de controles preventivos [26].

A nivel regulatorio, marcos como la EU AI Act y el AI Risk Management Framework (AI RMF) del NIST establecen la obligación de "gobernar, mapear, medir y gestionar" los riesgos de la IA. El NIST.AI.600-1 enfatiza que la GenAI requiere un enfoque de gobernanza a medida. Las empresas que no puedan demostrar trazabilidad y control sobre su código generado por IA podrían enfrentar sanciones o barreras de mercado en jurisdicciones estrictas como la Unión Europea.

6.4. Análisis complementario

La investigación confirma que la ausencia de un marco de referencia estructurado es el obstáculo principal para la adopción segura de GenAI en el desarrollo de software. La tecnología ha avanzado más rápido que la gobernanza, creando una brecha que genera riesgos. Este marco ofrece una solución basada en la rigurosidad de la Arquitectura Empresarial:

1. Architecture Building Blocks (ABBs): Capacidades fundamentales que toda organización necesita

2. Independencia tecnológica: Los ABBs aíslan los requisitos de seguridad de la volatilidad de las herramientas.

3. Base científica: Integra hallazgos de Pearce, Rivera Ladino, OWASP y NIST

4. Pragmatismo: Ofrece un camino viable para organizaciones de todos los tamaños.

7. Solución propuesta

7.1. Contexto de la solución

Como se mencionó en la introducción de este documento, el contexto actual de transformación digital acelerada en las organizaciones las enfrentan a una presión creciente para implementar soluciones tecnológicas que reduzcan tiempos de desarrollo, optimicen recursos y respondan con agilidad a demandas del mercado. La adopción de Inteligencia Artificial Generativa (GenAI) y arquitecturas con base en agentes promete beneficios significativos pero que de igual manera las expone a riesgos críticos como vulnerabilidades de seguridad en código generado, falta de cumplimiento normativo (GDPR, SOC 2), las llamadas "alucinaciones" de IA (inexactitudes, inventos) que generan código erróneo y pérdida de control sobre activos de IA entre otros.

El análisis de la literatura técnica, estándares industriales emergentes, y referentes tecnológicos como Gartner [27], McKinsey [28] entre otros, sugiere claramente que el éxito en la adopción de GenIA no reside en la implementación de los modelos (LLMs) más potentes o con mayor aceptación, sino en la madurez y robustez de la Arquitectura Empresarial y marcos de seguridad que los respalde y gobierne.

7.2. Descripción de la solución

Como se ha mencionado, actualmente no existe un framework o modelo de referencia único, estandarizado y ampliamente aceptado que oriente el desarrollo seguro, eficiente y escalable de aplicaciones empresariales en entornos GenAI/agentes. Esta propuesta presenta un Framework de Architecture Building Blocks (ABBs) basado en TOGAF 9.2, NIST AI RMF e ISO/IEC 42001, que proporciona:

- 8 capas arquitecturales operativas que cubren todo el SDLC
- 46 ABBs específicos agrupados por responsabilidades
- Mapeo a herramientas concretas (SBBs) del mercado actual
- Alineación con estándares globales de gobernanza, seguridad y cumplimiento
- Enfoque agnóstico aplicable a cualquier tipo de aplicación empresarial

Proporcionando un marco de referencia arquitectural agnóstico que permita a organizaciones diseñar, implementar, operar y gobernar aplicaciones empresariales seguras, eficientes y escalables desarrolladas mediante GenAI/agentes, asegurando integridad de datos, cumplimiento normativo, seguridad y optimización de recursos.

7.3. Estructura general del marco de referencia

Como desarrollo de la solución, se mostrará inicialmente una vista general del marco de referencia diseñado y posteriormente se procederá a describir cada bloque con sus características principales

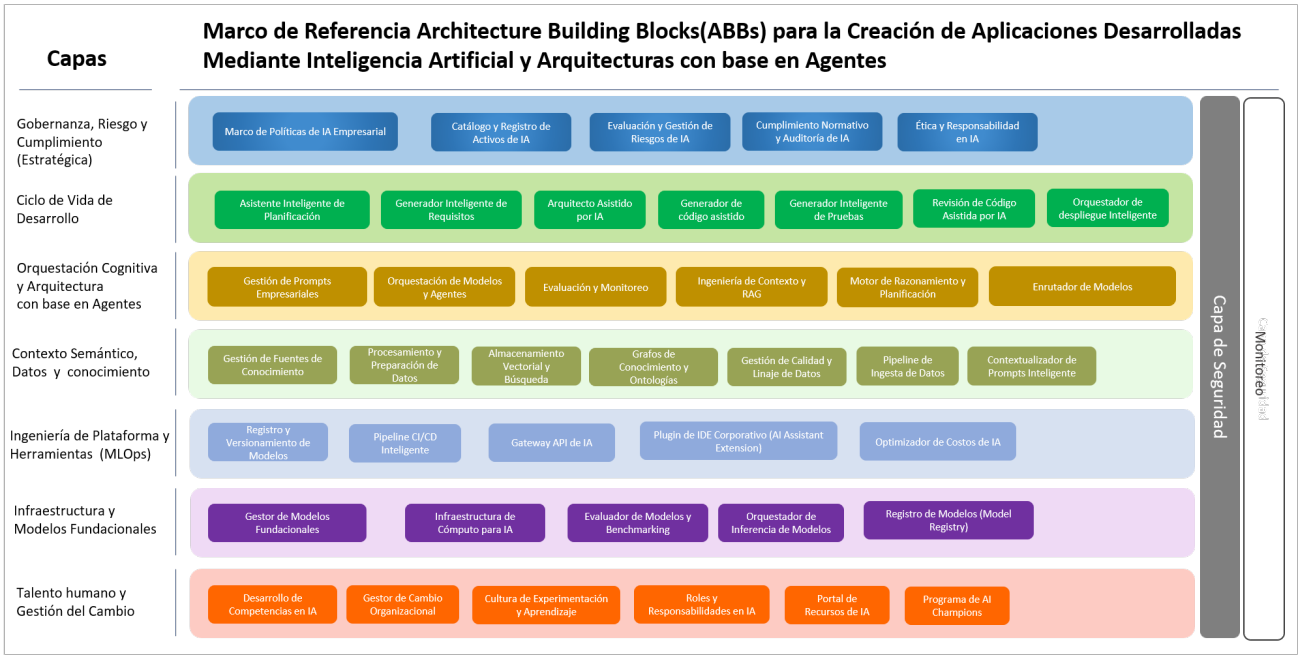


Figura 1: Blueprint / Vista General

7.4. Descripción de las capas

7.4.1. Capa 1: Gobernanza, Riesgo y Cumplimiento

Históricamente, la gobernanza en tecnología de la información ha operado como un mecanismo de control reactivo, a menudo burocrático y manual, que intervenía en puntos discretos del ciclo de vida del desarrollo. Sin embargo , en el contexto del desarrollo de software asistido por GenAI, donde la velocidad de producción de código y contenido supera en órdenes de magnitud la capacidad humana de revisión tradicional , este modelo colapsa. La arquitectura propuesta redefine la gobernanza, transformándola de un proceso administrativo a una capa de software activa y paralela al desarrollo mismo. [29]

Esta capa superior, de naturaleza estratégica y de control, tiene la función crítica de definir los límites operativos (de la inteligencia artificial antes, durante y después de la ejecución de cualquier proceso generativo. Su mandato principal es orquestar el cumplimiento normativo y la alineación ética, asegurando que la aceleración del desarrollo no vulnere los umbrales de riesgo de la organización. [21]Esto es particularmente vital en una Fintech, donde la trazabilidad y la auditabilidad no son opcionales, sino requisitos legales bajo marcos como la EU AI Act y regulaciones financieras locales.

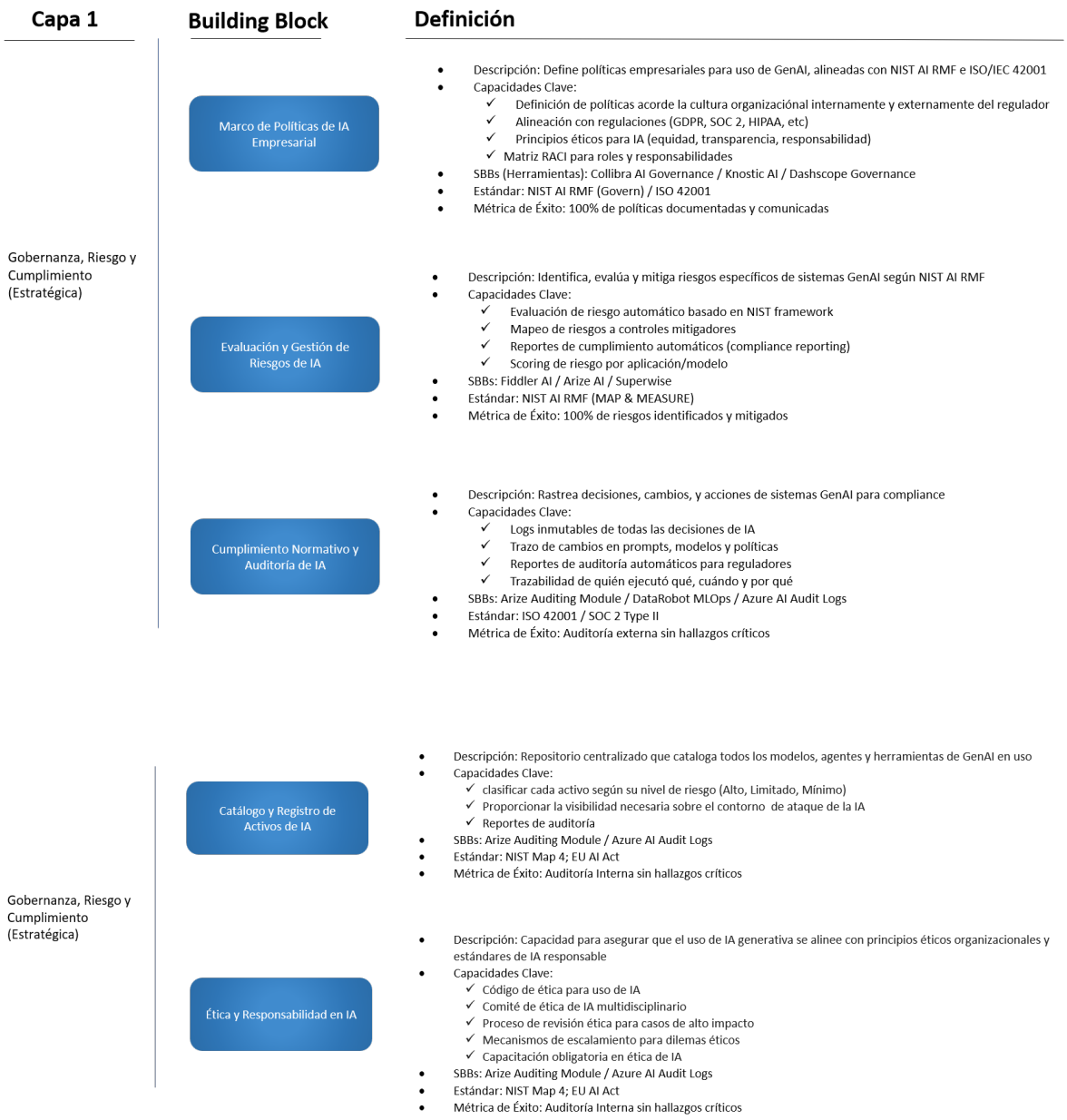


Figura 2: Capa 1 - Gobernanza, Riesgo y Cumplimiento (estratégica)

7.4.2. Capa 2: Ciclo de Vida de Desarrollo

La escalada de la Inteligencia Artificial Generativa en el ecosistema de desarrollo de software no representa únicamente una actualización o evolución de las herramientas de productividad [30], sino un cambio fundamental y profundo en la forma como lo veníamos haciendo en tecnología que redefine la naturaleza misma del Ciclo de Vida de Desarrollo de Software SDLC (Planeación, Annálisis, Diseño, Desarrollo, Pruebas, Implementación y Mantenimiento) acelerando cada fase sin comprometer control o calidad o por lo menos es justo eso lo que se busca. [31]

Capa 2	Building Block	Definición
Ciclo de Vida de Desarrollo	Asistente Inteligente de Planificación	• Descripción: Automatiza análisis de viabilidad, estimaciones y riesgos en fase de planificación • Capacidades Clave: ✓ Generación automática de project charters ✓ Análisis de riesgos predictivos basado en históricos ✓ Estimación inteligente de recursos y timeline ✓ Identificación de dependencias y camino crítico • SBBs: GitHub Copilot for Planning / Azure AI Planner / Amazon CodeWhisperer • Estándar: SDLC Agile/Waterfall / PMBOK • Métrica de Éxito: Estimaciones con margen de error < 15%
	Generador Inteligente de Requisitos	• Descripción: Extrae y refina requisitos de especificaciones de negocio y feedback • Capacidades Clave: ✓ Generación automática de historias de usuario ✓ Extracción de requisitos no funcionales (desempeño, seguridad, escalabilidad) ✓ Detección automática de gaps y conflictos en requisitos ✓ Generación de criterios de aceptación • SBBs: Claude for Requirements / GPT-4 API / Azure OpenAI • Estándar: IEEE 830 Standard for Software Requirements Specifications • Métrica de Éxito: 95%+ de requisitos completos y no-conflictivos
	Arquitecto Asistido por IA	• Descripción: Sugiere arquitecturas y patrones de diseño basados en requisitos y historias de usuario • Capacidades Clave: ✓ Recomendación de patrones arquitectónicos (microservicios, monolito, serverless) ✓ Generación automática de diagramas C4/UML ✓ Análisis de trade-offs de diseño (costo vs escalabilidad vs mantenimiento) ✓ Validación de arquitectura contra principios SOLID • SBBs: GitHub Copilot Design / Miro + GenAI / Draw.io AI Plugin • Estándar: C4 Model / UML 2.0 / TOGAF ADM • Métrica de Éxito: Arquitectura aprobada sin revisiones = 2 de cada 3
Ciclo de Vida de Desarrollo	Generador de código asistido	• Descripción: Genera código funcional según especificaciones y estándares de la IA generativa • Capacidades Clave: ✓ Generación de código multi-lenguaje (Python, Java, C#, JavaScript, Go) ✓ Aplicación automática de patrones corporativos y estándares ✓ Refactoring automático para mejorar legibilidad ✓ Generación de comentarios y documentación inline • SBBs: GitHub Copilot | Amazon CodeWhisperer | Tabnine • Estándar: PEP 8 / Google Style Guides / Corporate Standards • Métrica de Éxito: 40-60% de reducción en tiempo de codificación inicialmente
	Generador Inteligente de Pruebas	• Descripción: Crea casos de prueba unitarios, integración y E2E con alta cobertura • Capacidades Clave: ✓ Generación automática de casos de prueba desde código ✓ Datos de pruebas realistas basados en esquemas ✓ Optimización de cobertura de pruebas ✓ Generación de integración y pruebas end-to-end • SBBs: Codium AI / Diffblue Cover / GitHub Copilot Test Generation • Estándar: IEEE 1008 (Software Unit Testing) • Métrica de Éxito: Cobertura de código > 80% con generación automática
	Revisión de Código Asistida por IA	• Descripción: Capacidad para realizar análisis automatizado de calidad, seguridad y mejores prácticas • Capacidades Clave: ✓ Análisis estático de código (SAST) ✓ Detección de vulnerabilidades de seguridad ✓ Verificación de adherencia a estándares de código ✓ Identificación de deuda técnica ✓ Sugerencias de mejora automáticas • SBBs: SonarQube with AI |DeepSource | CodeRabbit | GitHub Copilot Pull Request Summaries • Estándar: IEEE 1008 (Software Unit Testing) • Métrica de Éxito: Cobertura revisión 100% con casos de éxito en hallazgos superior al 80%

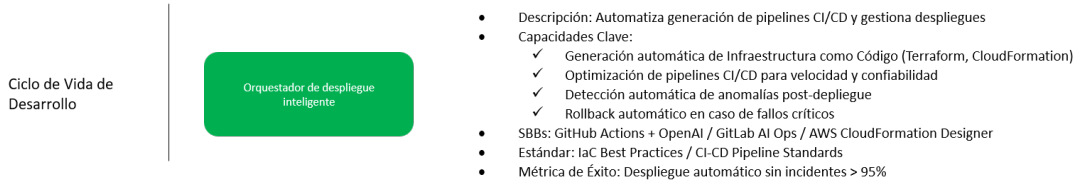


Figura 3: Capa 2 - Ciclo de Vida de Desarrollo

7.4.3. Capa 3: Orquestación Cognitiva y Arquitectura con base en Agentes

Esta capa gestiona la interacción directa con modelos de lenguaje masivos (LLMs) y la orquestación de múltiples servicios de IA para crear experiencias inteligentes, constituye el núcleo principal de las aplicaciones empresariales modernas potenciadas por GenIA, actuando como el intermediario inteligente entre los modelos de lenguaje masivos (LLMs) y la lógica de negocio. Esta capa es fundamental porque abstrae la complejidad inherente de trabajar con múltiples proveedores de IA (OpenAI, Anthropic Claude, cursor, etcétera) y permite una gestión centralizada de prompts, contextos y flujos conversacionales. [32][33]

El término Prompt

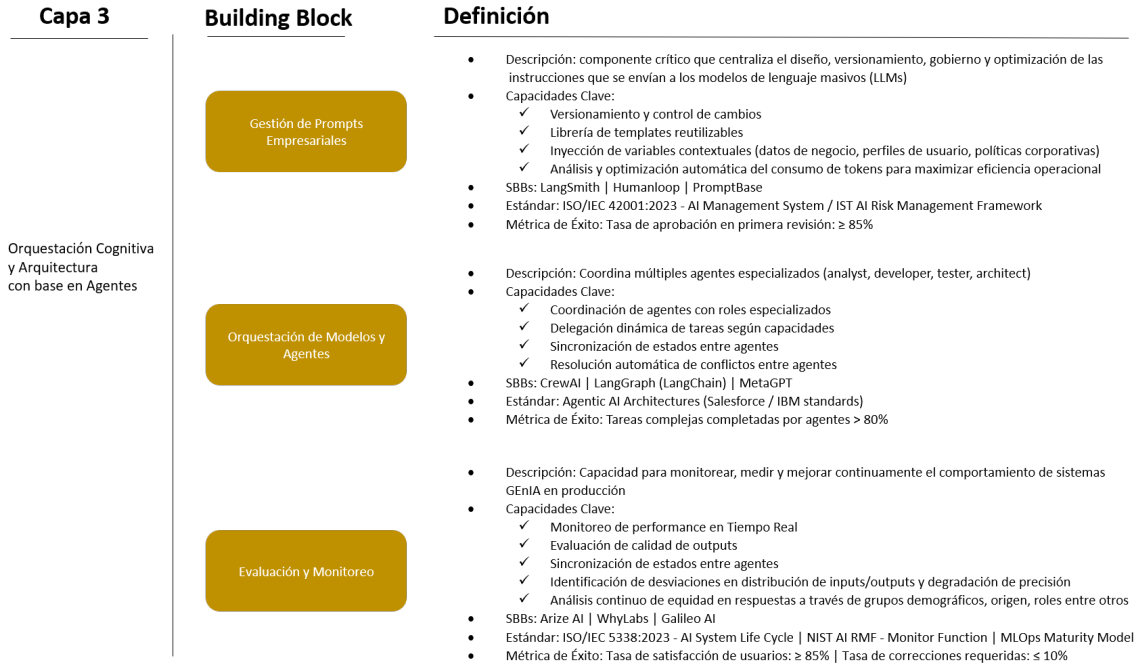


Figura 4: Capa 3 - Orquestación Cognitiva y Arquitectura con base en Agentes

7.4.4. Capa 4: Contexto Semántico, Datos y conocimiento

La principal limitación de los LLMs genéricos, independientemente de su tamaño, es su vacío respecto al contexto específico de la empresa, en ocasiones no tiene en cuenta o desconoce el código legado ("legacy"), las reglas de negocio privadas y la arquitectura interna de la organización. Esta capa tiene la responsabilidad de proporcionar el contexto necesario para "aterrizar" (grounding) las respuestas de la IA, reduciendo drásticamente las alucinaciones y asegurando la congruencia técnica con la estrategia de la empresa a nivel de datos e información. [34]

Capa 4	Building Block	Definición
Contexto Semántico, Datos y conocimiento	Gestión de Fuentes de Conocimiento	• Descripción: Crea índices vectoriales de documentos, código, estándares corporativos • Capacidades Clave: ✓ Indexación vectorial semántica de documentación ✓ Gestión de múltiples fuentes de conocimiento ✓ Actualización continua de índices ✓ Versionamiento de conocimiento • SBBs: Pinecone | Weaviate | Elasticsearch Vector Search • Estándar: Vector Database standards / RAG Architecture • Métrica de Éxito: Tiempo de indexación < 5 min para updates
	Procesamiento y Preparación de Datos	• Descripción: Capacidad para limpiar, transformar y estructurar datos no estructurados para consumo por modelos de IA. • Capacidades Clave: ✓ Extracción de texto de múltiples formatos (PDF, Word, HTML, código) ✓ Pipeline de validación y aseguramiento de calidad ✓ Construcción automática de grafos de conocimiento empresarial identificando entidades financieras (productos, clientes corporativos, contrapartes, reguladores), relaciones de negocio, etc ✓ Limpieza y normalización de texto • SBBs: Azure AI Document Intelligence / AWS Textract / Google Document AI • Estándar: ISO 8000 - Data Quality / DAMA-DMBOK (Data Management Body of Knowledge) / FAIR Principles • Métrica de Éxito: Calidad intrínseca de datos procesados de acuerdo a la tolerancia de la compañía
	Almacenamiento Vectorial y Búsqueda	• Descripción: Capacidad para almacenar y buscar eficientemente representaciones vectoriales de documentos y conocimiento. • Capacidades Clave: ✓ Índice vectorial con búsqueda por similitud (ANN) ✓ Filtrado por metadata ✓ Búsqueda híbrida (vectorial + keyword) ✓ Escalabilidad a millones de vectores ✓ CRUD operations sobre vectores • SBBs: • Pinecone / ChromaDB/ Weaviate • Estándar: RAG Architecture • Métrica de Éxito: Tiempo de búsqueda inferior en 30%
	Grafos de Conocimiento y Ontologías	• Descripción: Capacidad para representar conocimiento estructurado mediante grafos y ontologías para razonamiento avanzado. • Capacidades Clave: ✓ Modelado de entidades y relaciones ✓ Consultas sobre grafos (SPARQL, Cypher) ✓ Inferencia lógica sobre relaciones ✓ Integración con RAG (GraphRAG) ✓ Versionamiento de grafos • SBBs: Neo4j / Amazon Neptune / Microsoft GraphRAG • Estándar: GraphRAG
Contexto Semántico, Datos y conocimiento	Gestión de Calidad y Linaje de Datos	• Descripción: Capacidad para asegurar calidad de datos y rastrear su linaje desde el origen hasta su consumo • Capacidades Clave: ✓ Métricas de calidad de datos (completitud, precisión, actualidad) ✓ Seguimiento de linaje de datos ✓ Detección de anomalías en datos ✓ Validación de datos de entrenamiento ✓ Proceso de remediación de errores de calidad • SBBs: Apache Atlas / AWS Glue Data Quality / Monte Carlo Data • Estándar: IEC 25024:2015 - Data Quality Model / PMBOK
	Pipeline de Ingesta de Datos	• Descripción: Procesos automatizados (ETL para IA) que extraen, limpian, anonimizan y vectorizan continuamente la información de repositorios (Git, Confluence, Jira) • Capacidades Clave: ✓ Conectores para sistemas corporativos (APIs, databases, file systems) ✓ Programación de ingesta (batch, real-time) ✓ Manejo de errores ✓ Monitoreo de pipelines ✓ Actualizaciones incrementales • SBBs: Apache NiFi / Airbyte/ Fivetran • Estándar: Data Quality Model / PMBOK

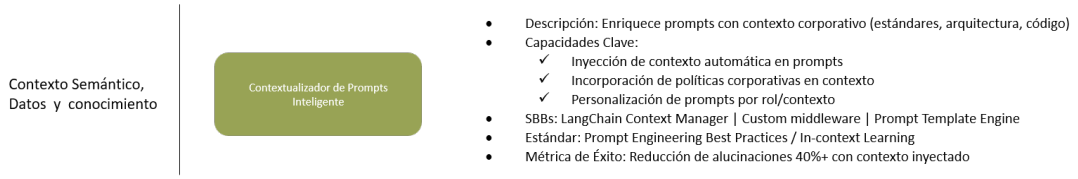
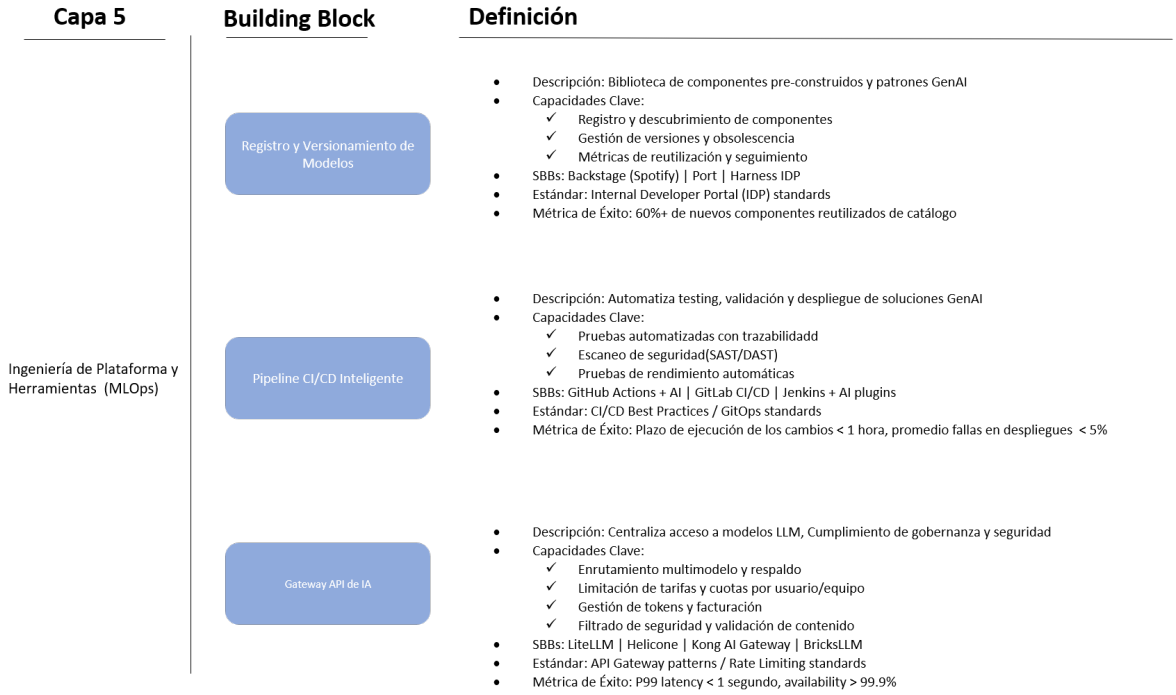


Figura 5: Capa 4 - Contexto Semántico, Datos y conocimiento

7.4.5. Capa 5: Ingeniería de Plataforma y Herramientas (MLOps)

Esta capa representa la "fábrica" donde se ensamblan y despliegan las soluciones de software. Siguiendo los principios modernos de Ingeniería de Plataforma, el objetivo es construir una Plataforma Interna de Desarrollo (IDP - Internal Developer Platform) [35] que abstraiga la complejidad subyacente de los modelos y agentes, ofreciendo capacidades de GenAI como servicios consumibles y sin fricción. [36]



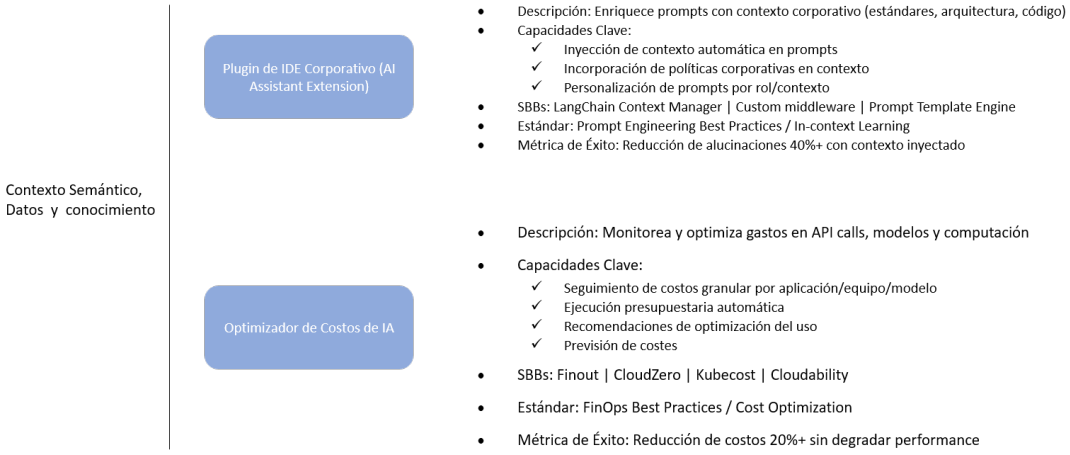
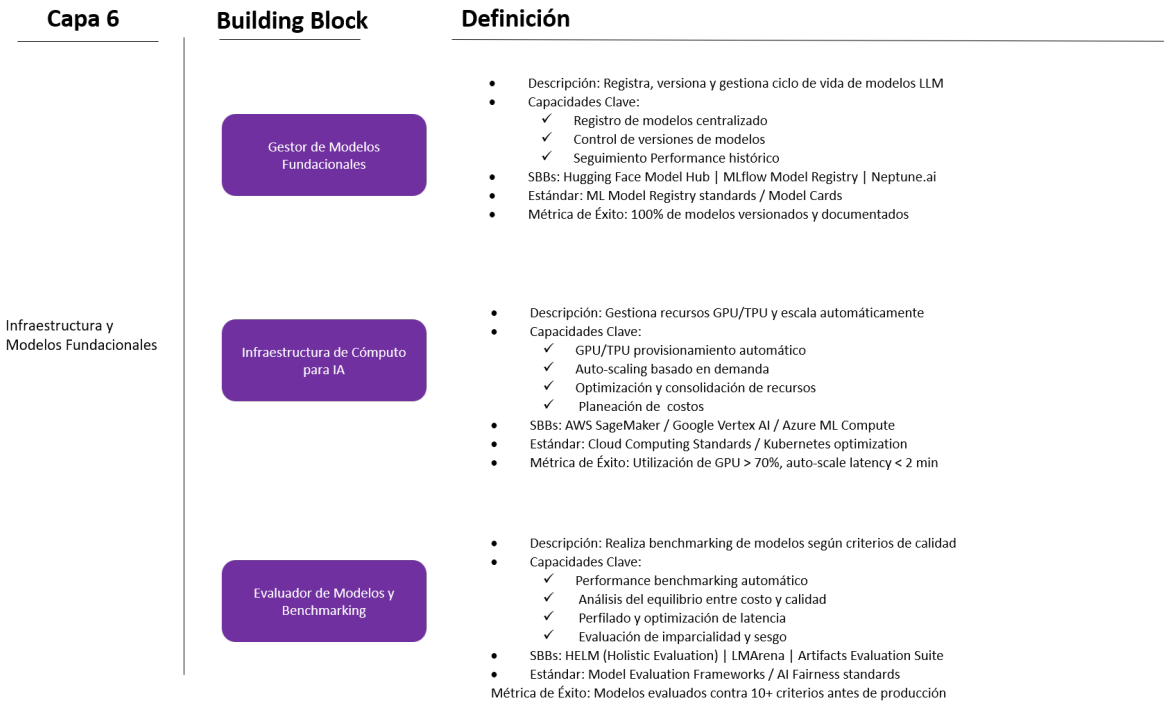


Figura 6: Capa 5 - Ingeniería de Plataforma y Herramientas (MLOps)

7.4.6. Capa 6: Infraestructura y Modelos Fundacionales

Esta capa constituye el sustrato físico y virtual que soporta toda la arquitectura superior [37]. Incluye los recursos de cómputo (GPUs, TPUs), los servicios en la nube y, crucialmente, los Modelos Fundacionales (LLMs y SLMs) mismos. La decisión arquitectónica clave que define esta capa es el equilibrio estratégico entre modelos propietarios consumidos como servicio (SaaS, e.g., OpenAI vía Azure) y modelos de código abierto alojados internamente [38].



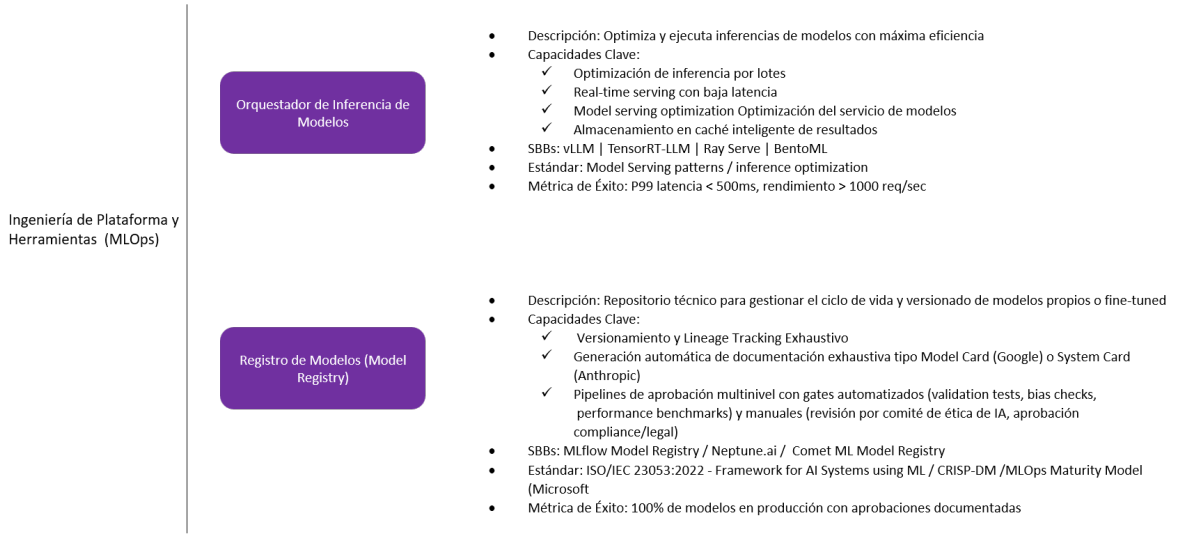


Figura 7: Capa 6 - Infraestructura y Modelos Fundacionales

7.4.7. Capa 7: Talento Humano y Gestión del Cambio

Más allá de los componentes técnicos, el despliegue de esta arquitectura implica una transformación radical en la estructura y cultura del talento humano. La tecnología es el habilitador, pero las personas son el vector de riesgo y éxito, se sabe que la tecnología sin adopción humana falla. [39] Esta capa crítica asegura que la organización tenga las habilidades , cultura y procesos para adoptar efectivamente GenAI, además de gestionar el upskilling, cambio cultural y ética. Se considera una capa crítica puesto que se estima que el 70 % de fracasos en los proyectos vienen de factor humano.

Contrario a la intuición de que la IA permite el uso de talento menos cualificado, ahora el talento se centra en la función de "Reviso" que requiere un entendimiento más profundo y sistémico de la tecnología para detectar errores sutiles o de lógica que la IA pueda introducir lo que pro supuesto demanda planes de formación agresivos en arquitectura de software y seguridad entre muchos otros. [40]

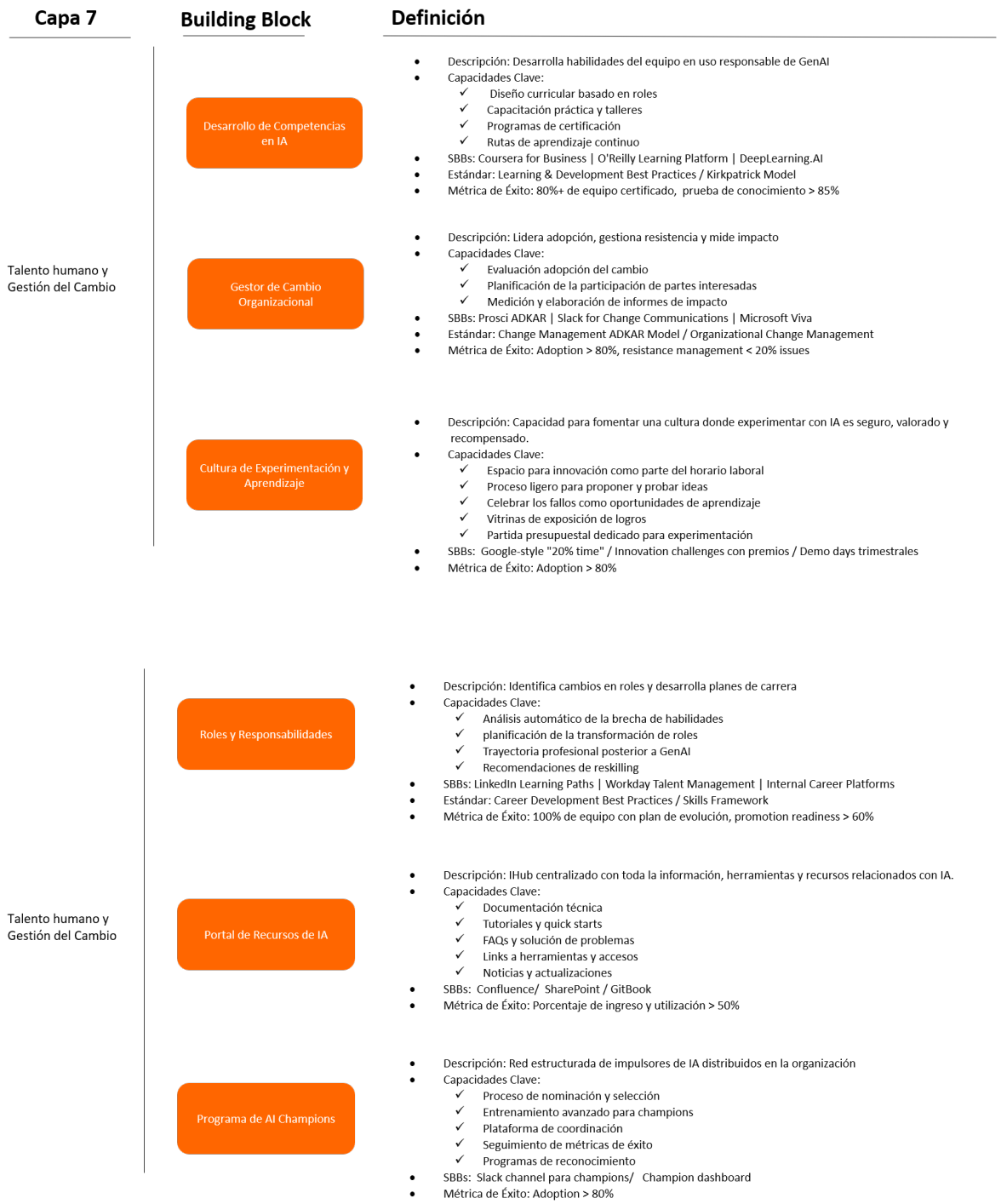


Figura 8: Capa 7 - Talento Humano y Gestión del Cambio

7.4.8. Capa Transversal de Seguridad y Defensa

La seguridad en la era de la GenAI no puede ser una etapa aislada al final del proceso o tenerse en cuenta solo como cumplimiento regulador o de principios de tecnología, sino que

debe ser una capa transversal que involucre todas las capas y cada unos de los componentes referidos en el marco de referencia. El panorama de amenazas en este contexto ha evolucionado en nuevas formas, introduciendo vectores de ataque inexistentes en el software tradicional, tales como el Prompt Injection (manipulación del comportamiento del modelo mediante entradas maliciosas), el Data Poisoning (corrupción de datos de entrenamiento/contexto para alterar decisiones) y las Supply Chain Vulnerabilities específicas de IA (como la alucinación de paquetes de software inexistentes que los atacantes pueden registrar para inyectar malware). [41]

Es por tanto una capa que por sí misma requiere un abordaje mucho más profundo que el contemplado en este documento de investigación, en todo caso se propone utilizar el concepto teórico basado en el principio de "Zero Trust Architecture" aplicado a IA generativa, donde cada interacción debe ser verificada y cada componente debe operar bajo el principio de mínimo privilegio [42].

Como una introducción a este abordaje presentamos un listado de componentes comunes que pueden servir de base para su implementación:

ID ABB	Nombre del Bloque (ABB)	Descripción Funcional y Justificación Técnica	Fuente / Estándar
ABB-SEC-01	Firewall de Prompts (Prompt Firewall)	Componente de inspección profunda de tráfico que analiza las entradas (prompts) y salidas del modelo en tiempo real. Busca patrones de intentos de inyección (jailbreaking), fuga de datos (DLP) o contenido malicioso. Tiene la capacidad de bloquear la transacción antes de que llegue al modelo o al usuario final	OWASP LLM01
ABB-SEC-02	Escáner de Vulnerabilidades de Cadena de Suministro AI	Herramienta especializada que verifica que las librerías, paquetes y dependencias sugeridas por la IA existan realmente en los repositorios oficiales y sean seguras. Mitiga ataques de "Package Hallucinati"n" o "Dependency Confusion", donde la IA sugiere importar código que no existe o es malicioso.	OWASP LLM05
ABB-SEC-03	Red Teaming Automatizado	Sistema que somete continuamente a los agentes y modelos a ataques simulados (adversarial testing) para identificar debilidades en sus respuestas, sesgos o fallos en los guardrails de seguridad. Permite una postura de seguridad proactiva y evolutiva frente a nuevos tipos de ataques a LLMs.	NIST Measure
ABB-SEC-04	Gestión de Identidad No Humana	Sistema robusto (IAM) para gestionar las credenciales, secretos y permisos de los agentes de IA. Asegura que un agente de codificación no tenga permisos excesivos ("Excessive Agency") que le permitan, por ejemplo, borrar bases de datos de producción o desplegar código sin autorización explícita.	OWASP LLM08

Tabla 3: Catálogo de ABBs para la Seguridad Transversal

8. Planeación del Trabajo

8.1. Descomposición de actividades WBS

A continuación se muestra la estructura de desglose de trabajo del proyecto:

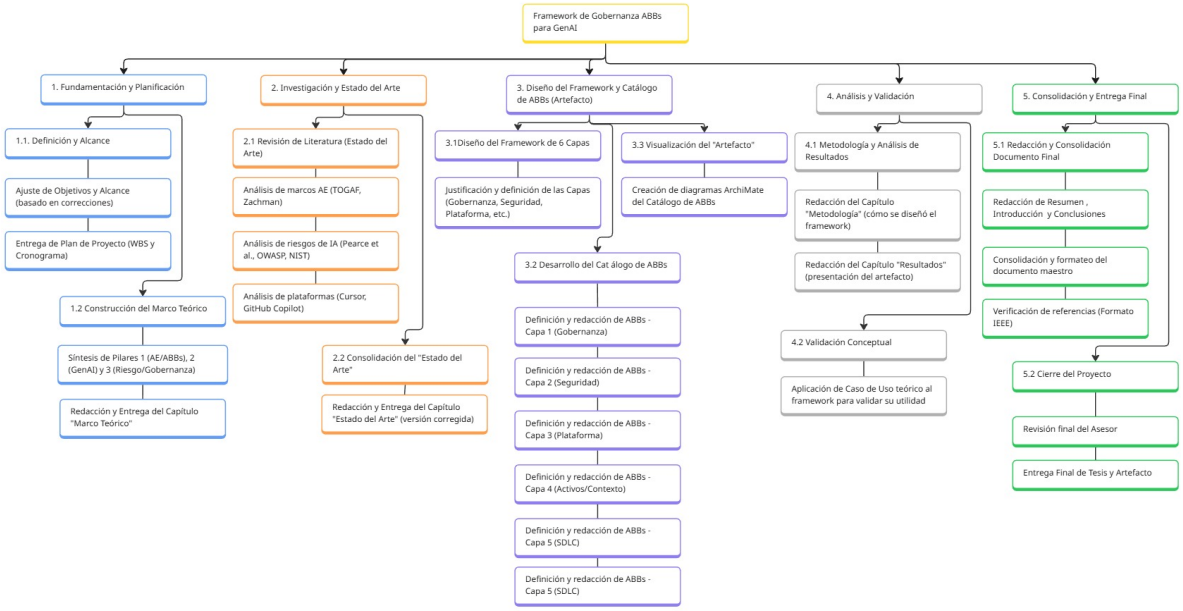


Figura 9: Descomposición de actividades - WBS

8.2. Diagrama de Gantt

La figura 10 muestra detalladamente la duración de cada Fase, actividad y tarea en el desarrollo de este proyecto.

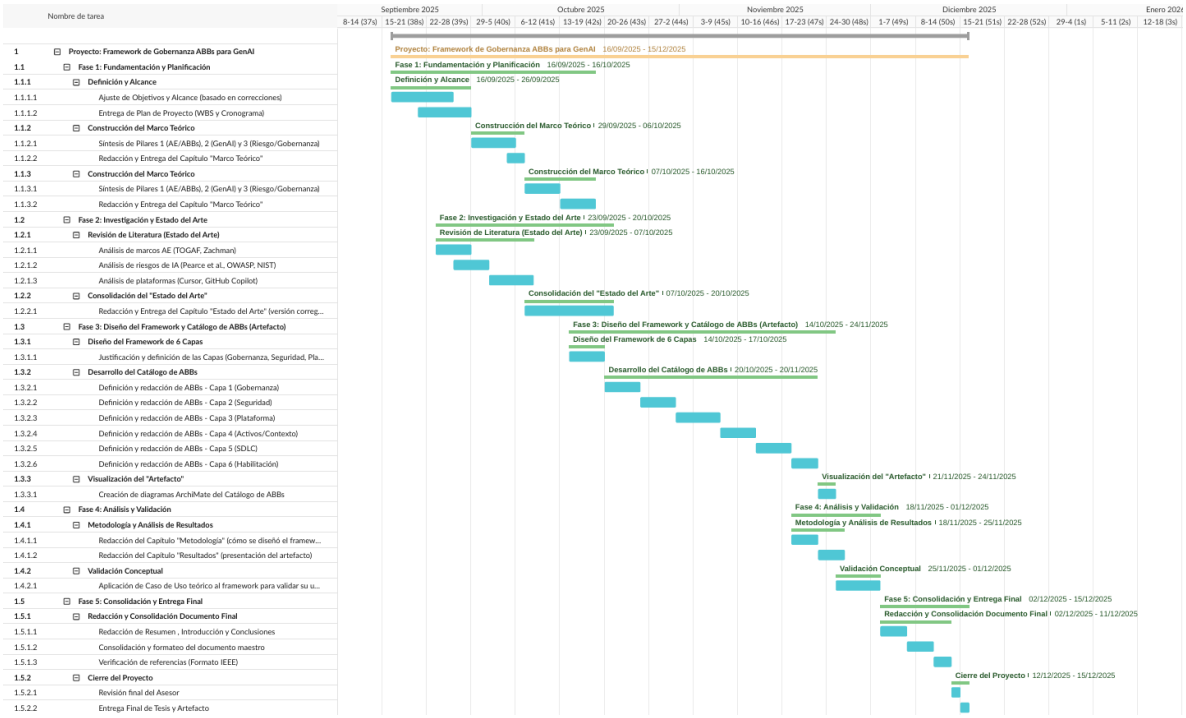


Figura 10: Diagrama de Gantt

9. Presupuesto

Con el objetivo de validar y retar el modelo de referencia propuesto se plantea un caso de uso común del cual se elaborará un presupuesto aproximado que nos permita evidenciar la utilidad del modelo pero a su vez la oportunidad en la reducción de complejidad, tiempo y por ende costo en la implementación de soluciones a partir de GenIA

Descripción General del Caso de Uso

El presente caso de uso consiste en el desarrollo e implementación de un sistema integral de gestión de Peticiones, Quejas, Reclamaciones (PQR), Incidencias y Emergencias para una institución financiera, operado a través de un portal web interno seguro. El sistema permite a clientes reportar problemas o solicitudes, las cuales son enrutadas automáticamente a través de tres niveles de soporte técnico especializado (primera, segunda y tercera línea), garantizando resolución eficiente, trazabilidad completa y cumplimiento normativo del sector financiero.

Concepto	Detalle / Rol / Servicio	Cantidad	Valor Unitario (USD)	Subtotal (USD)
Personas				
Arquitecto de IA/ABB	Diseño arquitectura y gobernanza	1	9,000/mes	9,000
Desarrollador Full Stack IA	Desarrollo frontend/backend asistido GenAI	2	6,000/mes	12,000
Prompt Engineer / QA	Ingeniería de prompts y validación GenAI	1	5,000/mes	5,000
Soporte y entrenamiento	Formación equipos en uso seguro IA	2	4,000/mes	8,000
Total Personas				34,000
SOFTWARE Y LICENCIAMIENTO				
Subscripción GenAI	GitHub Copilot, Azure OpenAI, Claude, etc.	5	120/mes	600
Ticketing/Workflow	Jira Service Mgmt o equivalente	10	30/mes	300
Testing Automático	CodiumAI, Diffblue, Selenium, etc.	2	150/mes	300
Seguridad DevSecOps	Snyk, Checkmarx, OWASP tools	1	500/mes	500
Total Software				1,700
SERVICIOS CLOUD				
Infraestructura Cloud	Compute/hosting/storage (AWS, Azure, GCP)	1	2,000/mes	2,000
GPU/IA On-Demand	Horas GPU/modelos IA	1	1,500/mes	1,500
API Mensual plataformas IA	LLM Gateway, Pinecone, Hugging Face, etc.	1	800/mes	800
Observabilidad y logs	Datadog, Grafana, New Relic	1	350/mes	350
Total Servicios Cloud				4,650
CAPACITACIÓN				
Cursos GenAI y SDLC IA	Coursera/Udemy/MS Learn	10	150/mes	1,500
Talleres seguridad	OWASP/Cybersecurity/ DevSecOps	5	200/mes	1,000
Total Capacitación				2,500
TOTAL INFRA Y OTROS				
Dominio interno web	Compra/gestión dominio SSL	1	200/mes	200
Backup y recuperación	Servicio en cloud	1	350/mes	350
Soporte técnico extra	Servicios 24/7 opcionales	1	500/mes	500
Total Infra/Otros				1,050
TOTAL GLOBAL				43,900

Tabla 4: Detalle del Presupuesto

10. Conclusiones

10.1. Hallazgos Principales y Cumplimiento de Objetivos

En el desarrollo de este trabajo de investigación se logró diseñar un framework o marco de referencia Architecture Building Blocks (ABBs) para el desarrollo seguro, eficiente y escalable de aplicaciones empresariales con Inteligencia Artificial Generativa y arquitecturas con base en agentes, cumpliendo así el objetivo general propuesto. Se identificaron, estructuraron y validaron 42 ABBs específicos distribuidos en 7 capas operativas más una capa transversal de seguridad, cada uno mapeado a herramientas del mercado (SBBs) y alineado con estándares internacionales: NIST AI RMF, ISO/IEC 42001, TOGAF 9.2 y OWASP.

Los resultados demuestran que el framework propuesto con bastante probabilidad ayudará a resolver la brecha identificada en el problema por la inexistencia de un marco base puesto que ahora se cuenta con un modelo único, estandarizado y operativo que guía el desarrollo de aplicaciones GenAI/agentes desde la gobernanza hasta la adopción organizacional. Los ABBs diseñados son agnósticos de industria, las fuentes de consulta provienen de empresas líderes (AWS, Microsoft, IBM, Salesforce) que garantiza la aplicabilidad y se puede adoptar a cualquier organización que tenga dentro de sus objetivos acelerar la adopción de IA con alcance empresarial

10.2. Contribuciones y aportes concretos

Los principales aportes de la investigación son :

1. Modelo integral: Primera taxonomía de 7 capas + 1 transversal que visibiliza el desarrollo GenAI/agentes.

2. ABBs Específicos y Operativos: 42 componentes arquitecturales concretos, cada uno con capacidades clave, SBBs mapeadas, estándares soportados y propuesta de métricas de éxito

3. Alineación con Estándares Globales: Se buscó integrar NIST AI RMF, ISO 42001, TOGAF y OWASP entre otros en un marco único de referencia.

4. Agnóstico y Transferible: Framework adaptable a cualquier industria (finanzas, salud, retail, gobierno) y stack tecnológico (cloud, on-prem, híbrido)

10.3. Limitaciones de la Investigación

Este estudio presenta limitaciones propias de la metodología de investigación:

1. Los resultados son con base en investigación pero aún no hay un ámbito corporativo que lo certifique y valide.

2. Aunque las fuentes principales son los estándares de la industria, también se utilizó investigaciones de empresas tecnológicas como McKinsey, Gartner, AWS que por su naturaleza comercial pueden tener una visión parcializada, que de todas maneras invita a seguir indagando y corroborando los resultados.

3. La vigencia del modelo puede afectarse por la velocidad de la innovación tecnológica en especial en cuanto a los SBBs propuestos.

4. La investigación tiene una profundidad técnica que excluye a personas que no tienen esta formación.

5. El alcance tecnológico de este modelo se orienta a empresas con alguna madurez tecnológica importante, para una empresa pequeña o sin foco en innovación y nuevas tecnologías, este modelo puede no ser el adecuado.

10.4. Recomendaciones para Implementación

Para la adopción e implementación de este modelo se sugiere:

1. Inicien con Capa 1, lo primero a resolver es la gobernanza, quién es responsable de qué.

2. Capacitación, si bien la IA facilita la generación de código, entender los resultados y lograr que estos sean exitosos requiere incluso un conocimiento más profundo para precisamente poder "controvertir" a la IA .

3. Piloto controlado: como toda tecnología nueva lo mejor es eligir un proyecto de bajo riesgo para validar el flujo end-to-end antes de escalar a sistemas críticos.

4. Lo que no se mide no se controla, se debe implementen desde el día uno métricas de calidad, seguridad, adopción y ROI para demostrar valor y rectificar de ser necesario.

5. Principio básico del agilismo, iterar lo más posible, es preferible avanzar paso a paso pero que cada paso sea seguro, desplegar, validar, ajustar y así cíclicamente .

Este proyecto de investigación no es solo una propuesta técnica sino una llamada a ser responsables, el boom de la inteligencia artificial y entre paréntesis su facilidad de ejecución está llevando a la exposición de riesgos crítico tanto de vulnerabilidades como de falta de valor y resultados favorables, la industria se encuentra en una encrucijada donde la velocidad de la IA puede generar innovación exponencial o crisis sistémica si no se gobierna adecuadamente. El framework ABBs propuesto espera demostrar que es tan importante avanzar como hacerlo con seguridad, y en tecnología eso significa disciplina arquitectural, liderazgo y una cultura de gobernanza desde el primer día.

Referencias

[1] M. de Tecnologías de la Información y las Comunicaciones – MinTIC, «Índice de Brecha Digital IBD 2022,» *https://colombiatic.mintic.gov.co/679/articles-333031_recurso_1.pdf*, 2023.

[2] Sothis, «Low Code: La revolución en el diseño de aplicaciones,» *https://www.sothis.tech/wp-content/uploads/2021/03/2007-EBook-APEB-LowCode.pdf*, 2021.

[3] Gartner, «Magic Quadrant for Enterprise Low-Code Application Platforms,» *Stamford, CT, USA: Gartner Inc*, 2023.

[4] H. Pearce, « Asleep at the Keyboard? Assessing the Security of GitHub Copilot's Code Contributions,» *in Proc. IEEE SP Workshops*, 2022.

[5] I. Forum, « Inteligencia Artificial y Ciberseguridad,» *Madrid, España: ISMS Forum España*, 2024.

[6] T. O. Group, « TOGAF® Standard, Version 9.2,» *https://www.opengroup.org/togaf-standard-version-92-overview*, 2018.

[7] J. C. Ledesma Alvear, *Frameworks de Arquitectura Empresarial*. Universidad Nacional de La Plata, 2017.

[8] A. Shah R. Srikant, *Generative AI for Developers*. Sebastopol, CA, USA. O'Reilly Media, 2023.

[9] J. M. R. Ladino, « Evaluación comparativa de vulnerabilidades de seguridad en código generado por modelos LLM,» *Bogotá, Colombia: Universidad Nacional de Colombia*, 2025.

[10] M. Brundage, « The Malicious Use of Artificial Intelligence: Forecasting, Prevention, and Mitigation,» *Oxford, UK: Oxford University*, 2020.

[11] R. Hernández Sampieri, C. Fernández Collado y P. Baptista Lucio, *Metodología de la investigación*, 6.ª ed. México: McGraw-Hill Education, 2014.

[12] Ministerio de Tecnologías de la Información y las Comunicaciones (MinTIC), «Déficit de talento digital en Colombia: Proyecciones 2025,» MinTIC, Bogotá, Colombia, Rep. MinTIC-2024-TD, 2024.

[13] O. Foundation, *OWASP top 10 for large language model applications version 1.1*, OWASP Foundation, [Online]. Available: https://owasp.org/www-project-top-10-for-large-language-model-applications/, Wakefield, MA, USA, oct. de 2023.

[14] O. Foundation, *OWASP top 10 for large language model applications version 2.0*, OWASP Foundation, [Online]. Available: https://genai.owasp.org/llm-top-10/, Wakefield, MA, USA, ago. de 2024.

[15] T. O. Group, «Architecture development method (ADM),» en *TOGAF® standard, version 9.2*, Reading, UK: The Open Group, 2018, ch. 5-14.

[16] T. O. Group, *TOGAF® standard, version 9.2*. Reading, UK: Van Haren Publishing, 2018, [Online]. Available: https://www.opengroup.org/togaf.

[17] S. Shah y S. Srikant, *Generative AI for Developers: Practical Applications for Building and Deploying LLM-Based Applications*. Sebastopol, CA, USA: O'Reilly Media, 2023.

[18] M. Malik, *Building production-ready RAG systems: Best practices and latest tools*, Medium, [Online]. Available: https://medium.com/@meeran03/building-production-ready-rag-systems-best-practices-and-latest-tools-581cae9518e7, mayo de 2024.

[19] Chitika, *Retrieval-augmented generation (RAG): 2025 definitive guide*, Chitika Blog, [Online]. Available: https://www.chitika.com/retrieval-augmented-generation-rag-the-definitive-guide-2025/, ene. de 2025.

[20] N. I. of Standards y Technology, «Artificial intelligence risk management framework (AI RMF 1.0),» U.S. Dept. Commerce, Gaithersburg, MD, USA, NIST AI 100-1, ene. de 2023. DOI: 10.6028/NIST.AI.100-1.

[21] I. O. for Standardization e I. E. Commission, *ISO/IEC 42001:2023 Information technology — Artificial intelligence — Management system*, [Online]. Available: https://www.iso.org/es/norma/42001, Geneva, Switzerland, 2023.

[22] H. Pearce et al., «Asleep at the keyboard? Assessing the security of GitHub Copilot's code contributions,» en *Proc. IEEE Symp. Security Privacy (SP)*, San Francisco, CA, USA, mayo de 2022, págs. 754-768. DOI: 10.1109/SP46214.2022.00057.

[23] E. Rivera Ladino, «Análisis comparativo de vulnerabilidades en código generado por modelos de lenguaje masivos,» *Rev. Colombiana Ing. Software*, vol. 3, n.º 1, págs. 45-67, ene. de 2025, [Online]. Available: http://revista.acis.org.co.

[24] Cloudflare, *What are the OWASP top 10 risks for LLMs?* Cloudflare Learning Center, [Online]. Available: https://www.cloudflare.com/learning/ai/owasp-top-10-risks-for-llms/, San Francisco, CA, USA, 2024.

[25] Securiti, *OWASP top 10 for LLM applications - complete guide*, Securiti Blog, [Online]. Available: https://securiti.ai/owasp-top-10-for-llms/, jun. de 2024.

[26] M. Brundage et al., *Toward trustworthy AI development: Mechanisms for supporting verifiable claims*, arXiv preprint arXiv:2004.07213, [Online]. Available: https://arxiv.org/abs/2004.07213, abr. de 2020.

[27] I. Gartner, «Innovation guide for generative AI in trust, risk and security management,» Gartner Research, Stamford, CT, USA, Rep. G00805234, abr. de 2024.

[28] M. Company, *The state of AI in 2024: Generative AI's breakout year*, McKinsey Digital, [Online]. Available: https://www.mckinsey.com/capabilities/quantumblack/our-insights/the-state-of-ai, New York, NY, USA, ago. de 2024.

[29] N. I. of Standards y Technology, «Artificial Intelligence Risk Management Framework (AI RMF 1.0),» NIST, inf. téc. NIST AI 100-1, ene. de 2023, [Online]. Available: https://nvlpubs.nist.gov/nistpubs/ai/NIST.AI.100-1.pdf.

[30] A. W. Services, *Transforming the Software Development Lifecycle (SDLC) with Generative AI*, AWS Architecture Blog, [Online]. Available: https://aws.amazon.com/architecture, ene. de 2025.

[31] M. Company, *How an AI-enabled software product development fuels innovation*, McKinsey Digital, [Online]. Available: https://www.mckinsey.com/digital, feb. de 2025.

[32] Salesforce, *Enterprise Agentic Architecture and Design Patterns*, Salesforce Architects, [Online]. Available: https://architect.salesforce.com, dic. de 2024.

[33] I. Corporation, *What Is Agentic Architecture?* IBM Think Topics, [Online]. Available: https://www.ibm.com/think, mar. de 2025.

[34] M. Corporation, *Retrieval Augmented Generation (RAG) in Azure AI Search*, Microsoft Learn, [Online]. Available: https://learn.microsoft.com, oct. de 2025.

[35] A. W. Services, *Accelerating Generative AI Applications with a Platform Engineering Approach*, AWS Machine Learning Blog, [Online]. Available: https://aws.amazon.com/machine-learning, nov. de 2025.

[36] P. E. Community, *AI in Platform Engineering: From Automation to Predictive Analytics*, Platformengineering.com, [Online]. Available: https://platformengineering.org, abr. de 2025.

[37] H. Face, *Hugging Face Model Hub Documentation*, Hugging Face Docs, [Online]. Available: https://huggingface.co/docs, 2025.

[38] G. Cloud, *Vertex AI Model Garden*, Google Cloud Documentation, [Online]. Available: https://cloud.google.com/vertex-ai, 2025.

[39] Prosci, *AI Adoption: Driving Change With a People-First Approach*, Prosci Blog, [Online]. Available: https://www.prosci.com, ago. de 2025.

[40] I. Corporation, *Transforming Change Management with Responsible AI*, IBM Insights, [Online]. Available: https://www.ibm.com/insights, mayo de 2025.

[41] Checkmarx, *DevSecOps Best Practices in the Age of AI*, Checkmarx Learn Center, [Online]. Available: https://checkmarx.com, jun. de 2025.

[42] O. Foundation, *LLM Prompt Injection Prevention Cheat Sheet*, OWASP Cheat Sheet Series, [Online]. Available: https://cheatsheetseries.owasp.org, ago. de 2023.