

MÁSTER EN FORMACION PERMANENTE EN INTELIGENCIA ARTIFICIAL

BusinessAnalystGPT

Presentado por:

JUAN MANUEL MIRON LUBIENIECKI

Dirigido por:

MIGUEL ANGEL TORRES FONT

CURSO ACADÉMICO 2024-2025

Resumen

BusinessAnalystGPT es un asistente de IA multiagente diseñado para automatizar la recolección y documentación de requisitos funcionales en proyectos de TI, reduciendo drásticamente los tiempos de entrega y elevando la calidad semántica mediante la integración de modelos de lenguaje a gran escala (LLMs), marcos de coordinación semántica y supervisión humana en el ciclo (*human-in-the-loop*). Este trabajo presenta sus fundamentos teóricos, metodología de diseño, implementación, evaluación y propuesta de negocio, destacando su potencial impacto en entornos empresariales.

Automatización rápida de requisitos basada en IA: BusinessAnalystGPT combina coordinación multiagente, generación aumentada con recuperación (RAG) y orquestación por roles para transformar procesos de especificación que antes tardaban semanas en tareas completadas en cuestión de horas, con mejoras de eficiencia de hasta el 80 %.

Fundamentos de sistemas multiagente (SMA): El proyecto se apoya en la teoría de SMA, enfatizando agentes autónomos descentralizados con roles diferenciados —consultores (LLMs), ejecutores (agentes especializados) y supervisores (humanos o híbridos humano-IA)— que cooperan mediante coordinación semántica y ontologías compartidas para una ejecución robusta de tareas.

Arquitecturas de agentes inteligentes: Se adopta el marco TB-CSPN (*Topic-Based Conditional Structured Process Networks*), que estructura la interacción entre agentes a través de tópicos semánticos, roles por capas y activación por umbrales, todo ello bajo supervisión humana jerárquica para preservar la intención semántica y el control ético.

Integración de LLMs como agentes activos: Los LLMs funcionan como agentes consultivos con capacidad de razonamiento, planificación y debate multiagente (MAD),

soportados por protocolos como AutoGen y Model Context Protocol (MCP) para acceder de forma segura a herramientas y datos externos, posibilitando una colaboración dinámica y contextual.

Mecanismos avanzados de coordinación: El sistema incorpora enfoques de razonamiento iterativo (ReAct) y debate multiagente (MAD) para mejorar la deliberación, la validación y la alineación semántica entre agentes, favoreciendo la transparencia y reduciendo errores en flujos de trabajo complejos.

Gestión de memoria y conocimiento: BusinessAnalystGPT emplea RAG y memoria semántica persistente (incluidas incrustaciones vectoriales y memoria episódica) para mantener la continuidad del contexto, la trazabilidad y la auditabilidad a lo largo de interacciones y tareas.

Enfoque metodológico: El desarrollo sigue una metodología experimental “Construir–Colaborar–Evolucionar”, que enfatiza la construcción iterativa de agentes, la colaboración multiagente mediada semánticamente, la coevolución competitiva para la mejora continua y una evaluación integral mediante *logging*, *tracing* y retroalimentación de usuarios.

Implementación y validación: El sistema se implementó con el framework AutoGen, seleccionado por su modularidad, soporte para flujos deliberativos multiagente e integración de los patrones SelectorGroupChat y Swarm para el control conversacional dinámico. Pruebas piloto con casos de uso reales evidenciaron viabilidad funcional, coherencia semántica y alta satisfacción de usuarios.

Modelo de negocio y potencial de mercado: Propuesto como solución SaaS B2B con planes escalonados, BusinessAnalystGPT se orienta a empresas de software y departamentos de TI en LATAM y Europa, atendiendo un mercado estimado de $\approx 650\,000$ organizaciones y millones de proyectos anuales con necesidad de análisis eficiente de requisitos.

Limitaciones y trabajo futuro: La tesis identifica restricciones como interacción multimodal limitada, dependencias de hardware para modelos locales grandes, integraciones empresariales aún incompletas y conjuntos de evaluación por ampliar. Se proponen líneas de investigación futura en refuerzo de memoria episódica, especialización por dominio, entradas multimodales y pruebas ampliadas en escenarios reales.

Palabras clave: sistemas multiagente, LLM, RAG, coordinación semántica, *human-in-the-loop*, AutoGen, MCP, ReAct, debate multiagente, especificación de requisitos.

Abstract

BusinessAnalystGPT is an advanced multiagent AI assistant designed to automate the collection and documentation of functional requirements in IT projects, dramatically reducing delivery times and improving semantic quality through integration of large language models (LLMs), semantic coordination frameworks, and human-in-the-loop supervision. This master's work presents its theoretical foundations, design methodology, implementation, evaluation, and business proposal, highlighting its potential impact in enterprise environments.

Rapid AI-driven requirements automation: BusinessAnalystGPT leverages multiagent coordination, retrieval-augmented generation (RAG), and role-based orchestration to transform weeks-long specification processes into tasks completed within hours, enhancing efficiency by up to 80%.

Multiagent system foundations: The project builds on multiagent system (MAS) theory, emphasizing decentralized autonomous agents with distinct roles—consultants (LLMs), executors (specialized agents), and supervisors (human or centauric)—cooperating through semantic coordination and shared ontologies for robust task execution.

Intelligent agent architectures: It adopts the TB-CSPN (Topic-Based Conditional Structured Process Networks) framework, which structures agent interaction via semantic topics, layered roles, and threshold-based activation guarded by hierarchical human supervision to preserve semantic intent and ethical control.

Integration of LLMs as active agents: LLMs function as consultative agents capable of reasoning, planning, and multiagent debate (MAD), supported by protocols such as AutoGen and Model Context Protocol (MCP) to access external tools and data securely, enabling dynamic, context-aware collaboration.

Advanced coordination mechanisms: The system incorporates iterative reasoning (ReAct) and multiagent debate (MAD) approaches to enhance deliberation, validation, and semantic alignment among agents, facilitating transparency and reducing errors in complex workflows.

Memory and knowledge management: BusinessAnalystGPT employs retrieval-augmented generation (RAG) and persistent semantic memory bases, including vector embeddings and episodic memory, to maintain context continuity, traceability, and auditability across interactions and tasks.

Methodological approach: The development follows a "Build–Collaborate–Evolve" experimental methodology, emphasizing iterative agent construction, semantically mediated multiagent collaboration, competitive co-evolution for continuous improvement, and comprehensive evaluation through logging, tracing, and user feedback.

Implementation and validation: The system was implemented using the AutoGen framework, chosen for its modularity, support for deliberative multiagent flows, and integration of SelectorGroupChat and Swarm patterns for dynamic conversational control. Pilot tests with real use cases demonstrated functional viability, semantic coherence, and high user satisfaction.

Business model and market potential: Proposed as a SaaS B2B solution with tiered subscription plans, BusinessAnalystGPT targets software companies and IT departments across LATAM and Europe, addressing a market of approximately 650,000 organizations and millions of annual projects needing efficient requirements analysis.

Limitations and future work: The thesis identifies constraints such as limited multimodal interaction, hardware restrictions for large local models, incomplete enterprise integrations, and evaluation benchmarks, outlining future research directions including episodic memory enhancement, domain specialization, multimodal inputs, and expanded real-world testing.

Keywords: multi-agent systems, LLMs, RAG, semantic coordination, human-in-the-loop, AutoGen, MCP, ReAct, multi-agent debate, requirements specification.

Indice General

Resumen.....	2
Abstract	5
1. Introducción	16
2. Marco Teórico.....	19
2.1 Fundamentos de los Sistemas Multiagente (MAS).....	22
2.1.1 Definición y Evolución Histórica	22
2.1.2 Tipologías de Agentes y Entornos	23
2.1.3 Problemas Clásicos de Coordinación.....	24
2.2 Inteligencia Artificial Agéntica.....	25
2.2.1 De la IA Simbólica a la IA Agéntica	25
2.2.2 Agentes Centáuricos y Monotónicos	26
2.2.3 Modelos Híbridos Humano-IA	27
2.3 Modelos de Coordinación Semántica	27
2.3.1 Ontologías Compartidas.....	27
2.3.2 Protocolos de Interacción Normativos.....	28
2.3.3 Mediación Semántica y Trazabilidad.....	28
2.4 Arquitectura TB-CSPN.....	29
2.4.1 Fundamentos Conceptuales	29
2.4.2 Capas Funcionales y Roles de Agentes	29

2.4.3 Umbrales de Activación y Supervisión Jerárquica	30
2.5 Redes de Petri en sistemas distribuidos	32
2.5.1 Petri Básicas vs. Coloreadas vs. Jerárquicas.....	32
2.5.2 Aplicaciones en Entornos Críticos.....	32
2.5.3 Comparación con otros Formalismos	33
2.6 Modelos de Lenguaje como Agentes (LLMs)	33
2.6.1 Evolución de los LLMs como Agentes.....	33
2.6.2 Capacidades Semánticas y Heurísticas	35
2.6.3 Limitaciones y Herramientas Complementarias	36
2.7 Herramientas en Arquitecturas Agenticas	36
2.7.1 Tipología de herramientas.....	36
2.7.2 Autogeneración y Adaptabilidad	37
2.7.3 Protocolos MCP y Estandarización	38
2.8 Enfoques Emergentes: ReAct y MAD	40
2.8.1 ReAct: Razonamiento Iterativo.....	40
2.8.2 MAD: Deliberación Multiagente	41
2.8.3 Integración en TB-CSPN	42
2.9 Modelado de Tópicos como Puente Semántico	43
2.9.1 LDA y Modelos Neuronales	43
2.9.2 Aplicaciones en Coordinación	44

2.9.3 Interpretabilidad y Control Semántico	45
2.10 Representaciones Externas y Memoria Organizacional	45
2.10.1 RAG y Recuperación Semántica	45
2.10.2 Diagramas, Logs y Trazabilidad	46
2.10.3 Memoria Persistente en Agentes Cognitivos	47
2.11 Conclusión del Marco Teórico.....	49
3. Metodología	50
3.1 Enfoque Metodológico General	52
3.2 Construcción del Agente.....	52
3.3 Colaboración Multiagente.....	53
3.4 Evolución del Sistema.....	54
3.5 Evaluación.....	55
3.6 Proceso Experimental e Implementación.....	56
3.6.1 Diseño Conceptual del Sistema y Definición de Roles de Agentes.....	56
3.6.2 Implementación Técnica del Agente BusinessAnalystGPT, Utilizando una Arquitectura Modular.....	61
3.6.3 Prueba piloto con casos de uso reales en entornos controlados.....	69
4. Resultados	3
4.1 Eficiencia Reducida de Semanas a Horas	4
4.2 Calidad Semántica y Capacidad Responsiva	5

4.2.1 Propósito	5
4.2.2 Evidencias extraídas del JSONL.....	5
4.2.3 Ejemplo ilustrativo (extracto JSONL)	6
4.2.4 Conclusión	7
4.2.5 Caso real “AI Dialer” - Presentación de la evidencia conversacional	8
4.2.6 Análisis de la traza de agentes: calidad semántica y seguimiento de preguntas	14
4.2.7 Caso real “AI Dialer” – Análisis funcional MAS de la traza	17
4.3 Nivel de Satisfacción del Usuario	23
4.3.1 Muestra y alcance	23
4.3.2 Instrumento (estructura y número de ítems)	23
4.3.3 Procedimiento de aplicación	24
4.3.4 Análisis	24
4.3.5 Trazabilidad con los productos evaluados	24
4.3.6 Conclusiones de los análisis de entrevistas y encuestas al panel experto	25
4.4 Diagrama y Gráfico: Calidad Funcional	27
4.5 Ahorro Global y Beneficios	28
4.6 Conclusión de Resultados	28
5. Propuesta de Negocio	29
5.1 Resumen Ejecutivo	29

	12
5.2 Problema para Resolver	29
5.3 Solución Propuesta.....	30
5.4 Propuesta de Valor	30
5.5 Modelo de Negocio	31
5.6 Mercado Objetivo	32
5.7 Competencia	32
5.8 Estrategia de Implementación.....	33
6. Conclusiones	35
7. Limitaciones y Futuras Líneas de Investigación.....	37
7.1 Limitaciones.....	37
7.2 Futuras Líneas de Investigación y Mejoras	38
Referencias.....	40
Acrónimos.....	48
Apéndice	50
Apéndice A	50
Apéndice B.....	51
Apéndice C.....	53
Apéndice D	56
Apéndice E.....	79

Figura 1	23
Figura 2	26
Figura 3	30
Figura 4	34
Figura 5	35
Figura 6	37
Figura 7	39
Figura 8	40
Figura 9	42
Figura 10	44
Figura 11	46
Figura 12	48
Figura 13	55
Figura 14	59
Figura 15	61
Figura 16	64
Figura 17	66
Figura 18	67
Figura 19	69
Figura 20	74
Figura 21	80
Figura 22	83

Figura 23	86
Figura 24	89
Figura 25	90
Figura 26	0
Figura 27	0
Figura 28	0
Figura 29	8
Figura 30	16
Figura 31	27

Tabla 1	31
Tabla 2	31
Tabla 3	77
Tabla 4	30
Tabla 5	31
Tabla 6	32
Tabla 7	34
Tabla 8	53
Tabla 9	55

1. Introducción

La inteligencia artificial (IA) atraviesa un punto de inflexión en el que el cambio ya no se mide año a año ni mes a mes, sino día a día. Cada jornada aparecen nuevos modelos fundacionales, “frameworks” de agentes y técnicas de optimización que amplían y a veces redefinen el estado del arte. La generación no se limita al texto: hoy conviven y evolucionan rápidamente capacidades multimodales de imagen, audio, vídeo y 3D; al mismo tiempo, emergen modelos de razonamiento (“reasoning models”) con cadenas deliberativas más profundas, y proliferan estrategias de modelos que aprenden de otros modelos (“distillation”, “self-play”, “synthetic data”), junto con optimizaciones de entrenamiento e inferencia que hacen viables casos de uso antes prohibitivos.

En paralelo, el paradigma aplicado ha mutado de asistentes conversacionales a agentes y, de ahí, a sistemas multiagente capaces de descomponer problemas, coordinarse y usar herramientas. Iniciativas como ChatDev (Qian et al., 2024) ilustran cómo agentes especializados, guiados por conversaciones estructuradas y mecanismos de verificación, colaboran para cubrir fases del ciclo de vida del software (diseño, codificación y prueba). De forma complementaria, (Hong et al., 2024) codifica procedimientos operativos estándar (SOPs) y salidas estructuradas entre roles (“Product”, “Architect, Engineer”, “QA”), reduciendo inconsistencias y elevando la calidad mediante “feedback” ejecutable durante la ejecución y depuración. Más allá de implementaciones concretas, distintas encuestas recientes ya proponen taxonomías metodológicas que conectan construcción, colaboración y evolución de agentes, y sistematizan retos de evaluación y despliegue en este campo tan dinámico (Luo et al., 2025).

Esta transición tecnológica exige también una relectura organizacional y humana: los sistemas más prometedores no son agentes aislados, sino colectivos heterogéneos que orquestan

personas, LLMs consultores y AIs especializadas mediante reglas de coordinación interpretables. En esa línea, diversos marcos formales modelan la formación dinámica de grupos por tópicos, con umbrales de participación, trazabilidad y capas de comunicación que facilitan la supervisión humana (o centáurica) y la alineación semántica entre roles. (TB-CSPN, 2025)

En este contexto de cambio vertiginoso y objetivos móviles, la investigación académica se enfrenta a dos retos prácticos: (i) diseñar artefactos que se beneficien de la frontera técnica sin quedar obsoletos en semanas, y (ii) evaluar de forma honesta sistemas que dependen de “stacks” y modelos en constante revisión. Esta memoria adopta una respuesta experimental y adaptativa: construir, integrar y medir un sistema real que haga de banco de pruebas de estas ideas.

Así, BusinessAnalystGPT es el vehículo de esta exploración: un asistente (multi)agente para automatizar la captura de requisitos funcionales a través de interacción natural, “retrieval-augmented generation” y orquestación basada en roles. El proyecto integra principios y componentes contemporáneos (p. ej., AutoGen para flujos multiagente, comunicación estructurada y memoria; orquestación por tópicos; trazabilidad semántica de decisiones) y se evalúa tanto en entornos controlados como en casos de uso reales, con foco en eficiencia, coherencia y experiencia de usuario.

Esta decisión metodológica se inspira en una idea sencilla, expresada por Bill Gates en reflexiones recientes: para aprender sobre IA en una época de evolución acelerada, hay que experimentar, usar lo más nuevo y “desconfundirse” aprendiendo en público leyendo, dialogando con expertos, probando sin miedo y estando dispuesto a evolucionar. En sus palabras, «el crecimiento de la IA avanza tan rápido que me ha sorprendido», lo que plantea si tendremos tiempo de adaptarnos; su consejo: mantenerse curioso, desarrollar pensamiento crítico y practicar la adaptabilidad. Este TFM se basa en esa filosofía: BusinessAnalystGPT es, en cierto sentido,

solo una excusa para aprender haciendo, iterar sobre lo último disponible y documentar con rigor qué funciona, qué no y por qué.

2. Marco Teórico

Este capítulo establece los fundamentos conceptuales, formales y arquitectónicos que sustentan el diseño de BusinessAnalystGPT. Su propósito es doble: (a) ofrecer un marco de referencia riguroso sobre sistemas multiagente (MAS) y IA agéntica incluida la incorporación de LLMs como agentes y (b) justificar, desde esa base, las decisiones de coordinación semántica, orquestación y trazabilidad que adopta la arquitectura propuesta (TB-CSPN). El recorrido avanza desde nociones clásicas (definición de agentes, problemas de coordinación, modelos formales) hacia enfoques recientes (ReAct, MAD, MCP, RAG, memorias persistentes), articulando un puente entre teoría y práctica (Ferber, 1999; Gruber, 1993; Hong et al., 2024; Lewis et al., 2020; Luo et al., 2025; Murata, 1989; Russell & Norvig, 2020; Wooldridge, 2009; Wu et al., 2023; Yao et al., 2022).

Propósito y alcance

Partimos de los fundamentos de los MAS autonomía, descentralización, heterogeneidad y coordinación y de su evolución histórica hasta arquitecturas donde modelos de lenguaje de gran tamaño actúan como agentes consultores o coordinadores. Sobre esa base, el capítulo caracteriza tipologías de agentes y entornos, y revisa problemas clásicos de coordinación (asignación, negociación, consenso, sincronización y comunicación semántica), con énfasis en los retos que emergen al integrar humanos, LLMs y agentes especializados (Ferber, 1999; Russell & Norvig, 2020; Wooldridge, 2009).

Ejes conceptuales del capítulo

IA agéntica: Se discute la transición de la IA simbólica a la IA agéntica, donde los LLMs asumen roles organizativos y responsabilidades dentro de sistemas distribuidos. Se introducen agentes centáuricos y el principio de monotonía semántica (prioridad del juicio

humano), y se describen modelos híbridos Humano–IA para colaboración segura y auditable (Borghoff et al., 2025; Luo et al., 2025).

Coordinación semántica: Se presenta el uso de ontologías compartidas, protocolos normativos de interacción y mecanismos de mediación y trazabilidad para garantizar interpretaciones coherentes entre agentes heterogéneos (Gruber, 1993; Hong et al., 2024; Wu et al., 2023).

Arquitectura TB-CSPN: Se formaliza una red de procesos estructurados condicionados por tópicos, organizada en capas funcionales (superficie, observación, cómputo), con umbrales de activación y supervisión jerárquica para preservar intención semántica y auditabilidad (Borghoff et al., 2025).

Modelos formales: Se justifica el empleo de redes de Petri (básicas, coloreadas y jerárquicas) para modelar concurrencia, sincronización y flujo de información con propiedades verificables (Jensen & Kristensen, 2009; Murata, 1989; Petri, 1962).

LLMs como agentes y herramientas: Se revisa la evolución de los LLMs hacia agentes consultores, sus capacidades semánticas y limitaciones, y la necesidad de herramientas complementarias (cálculo simbólico, acceso a datos, interacción estructurada). Se incluyen MCP como estándar de integración y ReAct/MAD como patrones de razonamiento y deliberación multiagente (Anthropic, 2024; Qin et al., 2023; Wu et al., 2023; Yao et al., 2022).

Representaciones y memoria organizacional: Se integran “topic modeling” (LDA y variantes neuronales) como puente semántico e interpretabilidad, y mecanismos de RAG y memoria persistente para continuidad contextual, recuperación verificable y control semántico (Blei et al., 2003; Lewis et al., 2020).

Articulación del capítulo

- 2.1–2.1.3: MAS, tipologías y problemas de coordinación.
- 2.2–2.2.3: IA agéntica, agentes centáuricos y modelos híbridos Humano–IA.
- 2.3–2.3.3: Coordinación semántica mediante ontologías, protocolos y trazabilidad.
- 2.4–2.4.3: Arquitectura TB-CSPN (capas y mecanismos de control).
- 2.5: Modelado formal con redes de Petri.
- 2.6–2.7: LLMs como agentes y capa de herramientas (incl. MCP).
- 2.8: ReAct y MAD como estrategias de razonamiento y debate multiagente.
- 2.9–2.10: Tópicos, RAG y memoria organizacional/persistente.
- 2.11: Síntesis e implicaciones para BusinessAnalystGPT.

Delimitaciones y supuestos

El capítulo se centra en arquitecturas híbridas con roles diferenciados y protocolos explícitos de interacción. No pretende ofrecer una revisión exhaustiva de todos los formalismos multiagente, sino seleccionar los pertinentes para sistemas centrados en semántica y trazabilidad. Se asume un entorno regulado donde la supervisión humana y la auditabilidad son requisitos, y donde los LLMs operan bajo controles organizativos (umbrales, normas y memoria compartida).

Contribución al proyecto

Este marco teórico fundamenta la elección de TB-CSPN y estructura los criterios de diseño y evaluación de BusinessAnalystGPT: coordinación por tópicos, supervisión jerárquica, protocolos normativos, uso de LLMs como consultores, herramientas estandarizadas (MCP), modelos formales (redes de Petri) y memoria organizacional (RAG + persistencia). Con ello, la tesis enmarca una solución auditable, interpretable y escalable para la captura de requisitos funcionales en contextos complejos y distribuidos.

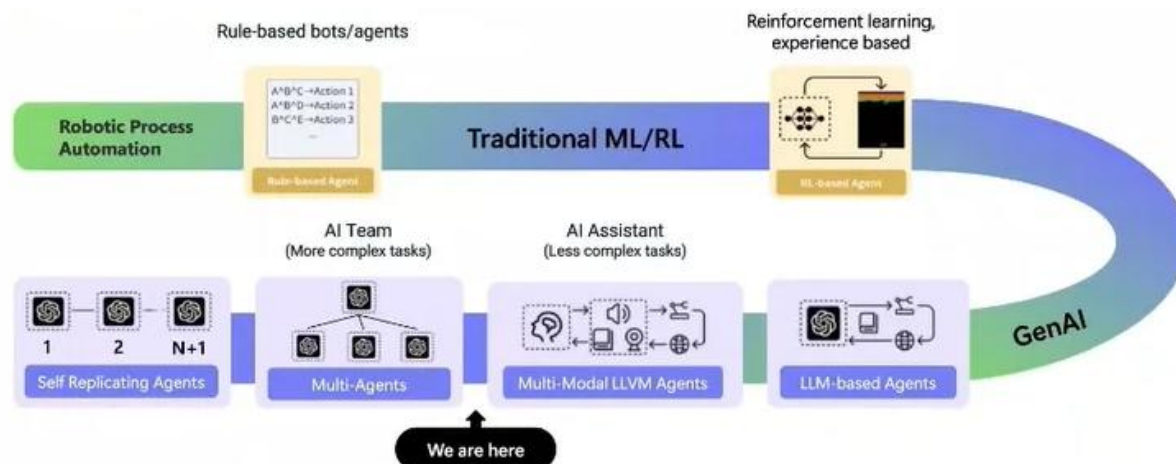
2.1 Fundamentos de los Sistemas Multiagente (MAS)

2.1.1 Definición y Evolución Histórica

Los sistemas multiagente (MAS, por sus siglas en inglés) constituyen una rama consolidada de la inteligencia artificial distribuida, centrada en el diseño, modelado y análisis de entornos compuestos por múltiples entidades autónomas que interactúan entre sí para alcanzar objetivos individuales o colectivos (Wooldridge, 2009). A diferencia de los sistemas monolíticos, los MAS se caracterizan por la descentralización del control, la heterogeneidad de los agentes involucrados y la necesidad de mecanismos de coordinación robustos.

Históricamente, el desarrollo de los MAS se remonta a los años ochenta, cuando comenzaron a surgir modelos computacionales inspirados en dinámicas sociales y organizacionales humanas (Ferber, 1999). Desde entonces, el campo ha evolucionado hacia arquitecturas cada vez más complejas, incorporando agentes con capacidades cognitivas avanzadas, razonamiento simbólico, aprendizaje automático y, más recientemente, modelos de lenguaje de gran tamaño (LLMs) como componentes activos del sistema (Luo et al., 2025).

Figura 1
Evolución de Agentes / MAS



(Saran, 2025)

2.1.2 Tipologías de Agentes y Entornos

En la literatura especializada, los agentes se definen como entidades autónomas capaces de percibir su entorno, razonar sobre él y actuar en consecuencia (Russell & Norvig, 2020). Esta definición ha dado lugar a múltiples tipologías, entre las que destacan:

- **Agentes reactivos:** responden directamente a estímulos del entorno sin planificación deliberada.
- **Agentes deliberativos:** construyen modelos internos del mundo y planifican sus acciones.
- **Agentes híbridos:** combinan componentes reactivos y deliberativos.
- **Agentes cognitivos:** incorporan capacidades de razonamiento simbólico, aprendizaje y adaptación.

Los entornos en los que operan los MAS también varían en complejidad, desde sistemas cerrados con reglas bien definidas hasta entornos abiertos, dinámicos y parcialmente observables. En estos últimos, la coordinación entre agentes se vuelve crítica, especialmente

cuando se integran agentes humanos, LLMs y sistemas especializados con capacidades dispares (Borghoff et al., 2025).

2.1.3 Problemas Clásicos de Coordinación

La coordinación en sistemas multiagente ha sido objeto de estudio intensivo, dado que la interacción entre agentes autónomos puede generar conflictos, redundancias o comportamientos no deseados si no se regula adecuadamente. Entre los problemas clásicos se encuentran:

- **Asignación de tareas:** cómo distribuir eficientemente las responsabilidades entre agentes.
- **Negociación:** mecanismos para resolver conflictos de interés.
- **Consenso:** alcanzar acuerdos en entornos distribuidos.
- **Sincronización:** garantizar que las acciones de los agentes se alineen temporalmente.
- **Comunicación semántica:** asegurar que los mensajes intercambiados sean interpretados correctamente por agentes con arquitecturas distintas.

La incorporación de LLMs como agentes consultores ha introducido nuevos desafíos, como la necesidad de mediación semántica, trazabilidad de decisiones y validación cruzada de inferencias (Wu et al., 2023; Hong et al., 2024). Estos retos han motivado el desarrollo de marcos organizacionales como TB-CSPN, que buscan estructurar la interacción entre agentes mediante capas funcionales, roles definidos y protocolos normativos.

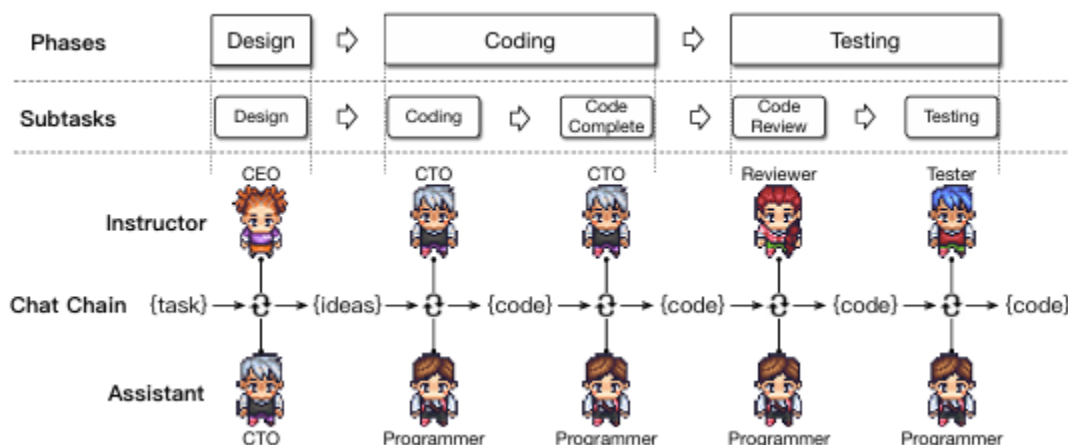
2.2 Inteligencia Artificial Agéntica

2.2.1 De la IA Simbólica a la IA Agéntica

La evolución de la inteligencia artificial (IA) ha transitado desde enfoques simbólicos, centrados en reglas explícitas y lógica formal, hacia modelos subsimbólicos basados en redes neuronales profundas. Sin embargo, la reciente integración de modelos de lenguaje de gran tamaño (LLMs) en sistemas multiagente ha dado lugar a una nueva categoría: la Inteligencia Artificial Agéntica. Este enfoque no se limita a la capacidad de procesamiento de los modelos, sino que los posiciona como entidades funcionales dentro de arquitecturas distribuidas, capaces de razonar, colaborar y tomar decisiones en interacción con otros agentes humanos y artificiales (Luo et al., 2025). La IA agéntica se caracteriza por la atribución de roles organizativos a los modelos de IA, permitiendo que participen activamente en procesos deliberativos, tareas de coordinación y validación semántica. Esta transición implica un cambio conceptual profundo: los modelos dejan de ser herramientas pasivas y se convierten en actores organizacionales con responsabilidades específicas dentro de sistemas complejos (Borghoff et al., 2025).

Figura 2

Ejemplo IA Agentica: Sistema Multi Agente con roles y responsabilidades



(Qian et al., 2024)

2.2.2 Agentes Centáuricos y Monotónicos

Uno de los conceptos más relevantes en este nuevo paradigma es el de los agentes centáuricos, definidos como híbridos funcionales compuestos por humanos asistidos por IA. Estos agentes operan bajo el principio de monotonía semántica, según el cual la IA puede enriquecer, clarificar o reforzar las decisiones humanas, pero nunca contradecirlas ni alterar su intención original (Borghoff et al., 2025). Este principio es especialmente crítico en entornos sensibles como la medicina, la seguridad o la toma de decisiones éticas, donde la preservación del juicio humano es prioritaria. La arquitectura TB-CSPN incorpora este principio mediante mecanismos como umbrales de control y supervisión jerárquica, que garantizan que las decisiones estratégicas permanezcan bajo control humano, mientras que los agentes IA contribuyen con velocidad, precisión y consistencia operativa.

2.2.3 Modelos Híbridos Humano-IA

La colaboración entre humanos y agentes IA requiere estructuras organizativas que regulen la interacción, minimicen errores y aseguren coherencia semántica. En este sentido, la IA agentica se apoya en modelos híbridos que combinan: juicio humano contextual, capacidades de los LLMs y agentes especializados. Estos modelos híbridos permiten una distribución funcional de roles, donde cada tipo de agente aporta desde su competencia específica, pero dentro de un marco organizativo común. La arquitectura TB-CSPN formaliza esta colaboración mediante capas funcionales (superficie, observación, cómputo) y protocolos normativos que estructuran la interacción entre agentes (Hong et al., 2024; Wu et al., 2023).

2.3 Modelos de Coordinación Semántica

2.3.1 Ontologías Compartidas

La coordinación semántica en sistemas multiagente requiere que los agentes involucrados compartan una comprensión común del entorno, las tareas y los conceptos clave. Para ello, se emplean ontologías compartidas, que actúan como marcos de referencia semánticos estructurados. Una ontología define formalmente los términos, relaciones y reglas que rigen un dominio específico, permitiendo que agentes con arquitecturas distintas, humanos, LLMs, agentes especializados interpreten los datos de manera coherente (Gruber, 1993).

En el contexto de TB-CSPN, las ontologías no solo facilitan la interoperabilidad entre agentes, sino que también permiten la trazabilidad semántica de las decisiones. Esto es especialmente relevante cuando se integran agentes centáuricos, cuya interpretación humana debe ser preservada y enriquecida por la IA sin distorsión conceptual (Borghoff et al., 2025).

2.3.2 Protocolos de Interacción Normativos

Más allá del vocabulario compartido, la coordinación efectiva requiere protocolos normativos de interacción, que regulen cómo los agentes se comunican, validan información, resuelven conflictos y escalan decisiones. Estos protocolos definen estructuras de diálogo, formatos de mensajes, reglas de validación y mecanismos de retroalimentación.

Modelos como AutoGen y MetaGPT han demostrado que la implementación de protocolos explícitos mejora la eficiencia, la coherencia y la auditabilidad de los sistemas multiagente (Wu et al., 2023; Hong et al., 2024). En TB-CSPN, estos protocolos se integran en las capas funcionales del sistema, permitiendo que cada tipo de agente opere dentro de un marco normativo adaptado a su rol y capacidades.

2.3.3 Mediación Semántica y Trazabilidad

La mediación semántica es el proceso mediante el cual se asegura que los conceptos clave sean interpretados de forma coherente por todos los agentes, incluso si sus mecanismos de procesamiento son distintos. En arquitecturas híbridas, esta mediación se realiza a través de tópicos compartidos, representaciones vectoriales y mecanismos de validación cruzada.

El marco TB-CSPN introduce el concepto de tópicos como unidades semánticas dinámicas, que encapsulan el significado contextual de una conversación, tarea o flujo de trabajo. Estos tópicos son gestionados por agentes consultores (LLMs), supervisados por agentes humanos y utilizados por agentes ejecutores para guiar sus acciones. Esta estructura permite una trazabilidad semántica completa, desde la intención estratégica hasta la ejecución operativa (Luo et al., 2025).

Además, se incorporan mecanismos como “threshold guards” y “event spaces”, que regulan la activación de transiciones en función del estado semántico del sistema. Esto garantiza

que las decisiones no solo sean técnicamente correctas, sino también alineadas con los valores y objetivos humanos.

2.4 Arquitectura TB-CSPN

2.4.1 Fundamentos Conceptuales

La arquitectura Topic-Based Conditional Structured Process Networks (TB-CSPN) representa una propuesta innovadora para la coordinación semántica en sistemas multiagente híbridos. Este marco conceptual parte de la premisa de que la interacción entre agentes humanos, LLMs y especializados debe estar mediada por estructuras organizativas que preserven la intención semántica, regulen la ejecución distribuida y faciliten la trazabilidad de decisiones (Borghoff et al., 2025).

A diferencia de los modelos tradicionales basados en grafos de comunicación o diagramas de estados, TB-CSPN introduce una abstracción organizativa basada en procesos estructurados condicionados por tópicos. Cada transición en el sistema está vinculada a un estado semántico representado por un tópico, cuya activación depende de umbrales definidos y supervisión jerárquica. Esta lógica permite modelar flujos de trabajo complejos, donde la semántica compartida guía la ejecución técnica.

2.4.2 Capas Funcionales y Roles de Agentes

La arquitectura se organiza en tres capas funcionales, cada una asociada a un tipo de agente:

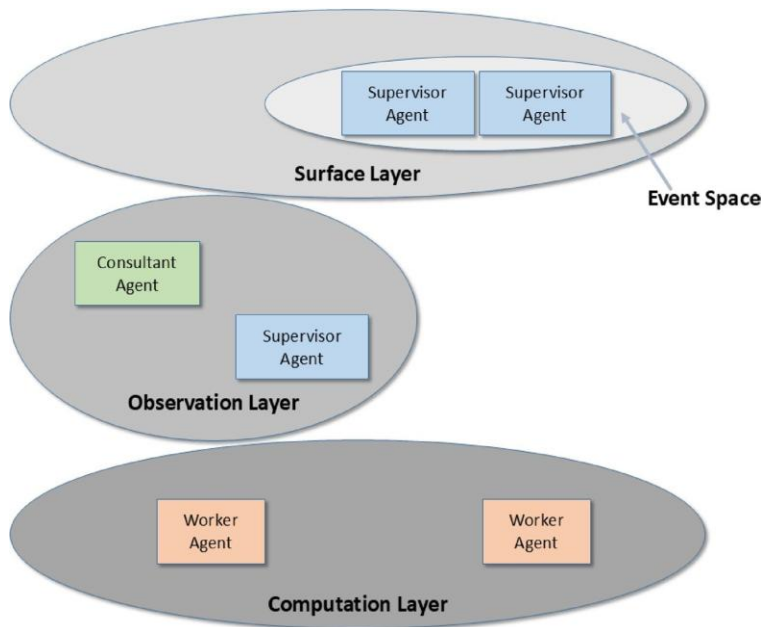
- **Capa de Superficie (“Surface Layer”):** donde operan los agentes supervisores (humanos y centáuricos), responsables de decisiones estratégicas y validación ética.
- **Capa de Observación (“Observation Layer”):** gestionada por agentes consultores (LLMs), encargados de interpretar, sintetizar y mediar semánticamente.

- **Capa de Cómputo (“Computation Layer”):** donde actúan los agentes ejecutores, especializados en tareas técnicas y deterministas.

Esta separación funcional permite una coordinación transversal, en la que los agentes pueden interactuar entre capas sin perder su especialización. Además, cada transición del sistema pertenece exclusivamente a una capa, lo que facilita la trazabilidad y el control semántico.

Figura 3

Estructura organizacional en TB-CSPN



(Borghoff et al., 2025)

2.4.3 Umbrales de Activación y Supervisión Jerárquica

Uno de los mecanismos clave de TB-CSPN es el uso de “threshold guards”, que definen condiciones semánticas para la activación de transiciones. Estos umbrales permiten regular el flujo de información y acciones en función del estado de los tópicos compartidos, evitando ejecuciones prematuras o desalineadas.

Complementariamente, se implementa una supervisión jerárquica, en la que los agentes humanos o centáuricos validan las decisiones críticas antes de su ejecución. Este modelo

garantiza que la semántica original definida por los humanos se mantenga intacta, mientras que los agentes IA contribuyen con eficiencia operativa.

La arquitectura también contempla el uso de espacios de eventos (“event spaces”) como interfaz entre estímulos externos y dinámicas internas, permitiendo que los agentes interpreten el contexto de manera coherente y reactiva.

En conjunto, TB-CSPN ofrece una solución robusta para la coordinación semántica en sistemas multiagente, integrando principios organizacionales, mecanismos de control y estructuras formales que permiten una colaboración efectiva entre humanos e inteligencia artificial.

Tabla 1

Mapeo de componentes TB-CSPN a implementaciones practicas

System	TB-CSPN Analog	Limitations
LangChain ReAct	Consultant-Worker interactions	Procedural, lacks semantic modeling
AutoGen MAD	Multi-agent Surface transitions	No layered semantics
TogetherAI Orchestrator	Supervisor-Worker split	Static role assignment, no adaptation
Agentic RAG	Topic tokens as Surface signals	Manual topic curation
ReWOO	Sequential Worker flows	Hardcoded logic, no dynamic roles

Tabla 2

Mapeo de las funciones de los agentes a través de las capas de comunicación

LAYER/ROLE	Supervisor	Consultant	Worker
Surface	Strategic direction; external mediation	Optional preprocessing (e.g., early filtering)	<i>Rare</i> : direct interaction with raw data
Observation	Guidance of semantic interpretation	Semantic analysis; topic refinement; heuristic reasoning	Support for translation into computation
Computation	Policy enforcement; value alignment	Transformation into operational tokens	Execution of domainspecific tasks

2.5 Redes de Petri en sistemas distribuidos

2.5.1 Petri Básicas vs. Coloreadas vs. Jerárquicas

Las redes de Petri constituyen una herramienta formal ampliamente utilizada para el modelado de sistemas concurrentes, distribuidos y asincrónicos. Introducidas por Carl Adam Petri en la década de 1960, estas redes se basan en grafos bipartitos compuestos por lugares (representan estados o condiciones) y transiciones (representan eventos o acciones), conectados mediante arcos dirigidos (Petri, 1962).

En el marco TB-CSPN, se emplea una variante avanzada conocida como redes de Petri coloreadas, que extienden el modelo clásico permitiendo que los tokens transporten datos diferenciados (colores). Esta capacidad es esencial para representar distribuciones complejas de tópicos semánticos, ya que cada token puede codificar un estado contextual específico, influenciando el comportamiento de las transiciones en función del contenido semántico (Jensen & Kristensen, 2009).

Además, se contempla el uso de redes de Petri jerárquicas, que permiten modularidad, encapsulación y activación condicional. Estas propiedades son fundamentales para modelar sistemas multiagente con múltiples capas funcionales, como los propuestos en TB-CSPN, donde cada capa puede tener subprocesos internos regulados por condiciones semánticas específicas.

2.5.2 Aplicaciones en Entornos Críticos

La elección de redes de Petri como base formal para TB-CSPN responde a su capacidad para garantizar propiedades como vivacidad, accesibilidad, acotación y ausencia de bloqueos. Estas propiedades son especialmente relevantes en entornos críticos como salud, finanzas o

seguridad, donde la trazabilidad y la verificabilidad de las decisiones son requisitos normativos (Murata, 1989).

A diferencia de otros formalismos como las álgebras de procesos (por ejemplo, π -cálculo), que, aunque precisos a nivel lógico son menos intuitivos para capturar estructuras organizacionales, las redes de Petri ofrecen una representación visual clara y una semántica operacional directa. Esto facilita tanto la implementación como la auditoría de sistemas híbridos humano-IA.

2.5.3 Comparación con otros Formalismos

En comparación con modelos basados en máquinas de estados finitos, las redes de Petri permiten representar concurrencia explícita, sincronización condicional y flujo de tokens, lo que las hace más adecuadas para sistemas distribuidos. Además, su capacidad para integrar datos semánticos mediante tokens coloreados las posiciona como una herramienta ideal para arquitecturas como TB-CSPN, donde la semántica compartida es el eje de la coordinación.

La posibilidad de asignar transiciones a decisiones humanas, por ejemplo, estratégicas, éticas o contextuales permite modelar el juicio humano sin necesidad de representar su cognición interna. Esta propiedad convierte a las redes de Petri en un puente formal entre la lógica computacional y la intención humana, facilitando la colaboración efectiva en sistemas híbridos (Borghoff et al., 2025).

2.6 Modelos de Lenguaje como Agentes (LLMs)

2.6.1 Evolución de los LLMs como Agentes

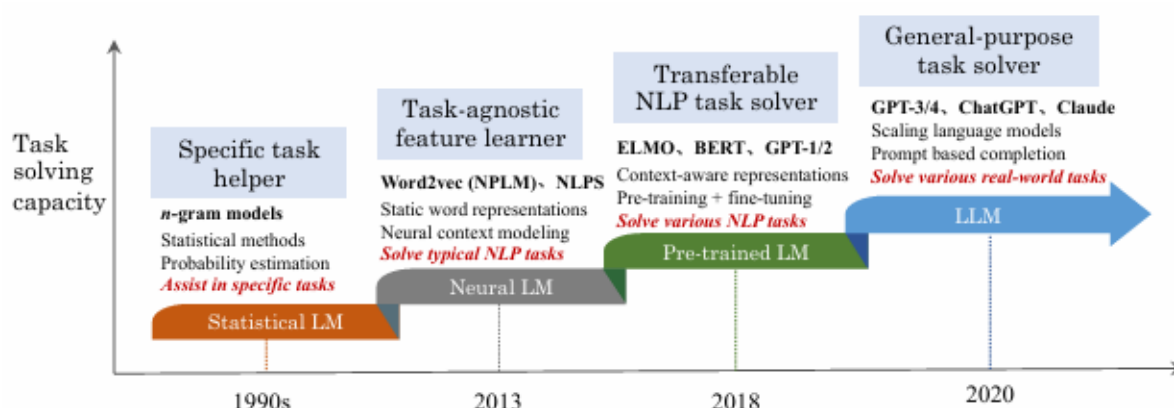
Los modelos de lenguaje de gran tamaño (LLMs) han evolucionado rápidamente desde sistemas de generación textual hacia entidades funcionales capaces de participar activamente en

procesos de razonamiento, planificación y coordinación. Esta transformación ha sido documentada por Luo et al. (2023), quienes destacan el surgimiento de los “LLM Agents” como una nueva categoría dentro de la inteligencia artificial agéntica.

En este nuevo rol, los LLMs no solo procesan lenguaje natural, sino que también actúan como agentes consultores, capaces de interpretar contextos, sintetizar información y colaborar con otros agentes humanos o artificiales. Esta capacidad se ha potenciado mediante arquitecturas como AutoGen y MetaGPT, que permiten a los LLMs interactuar mediante protocolos normativos, acceder a herramientas externas y participar en flujos de trabajo multiagente (Wu et al., 2023; Hong et al., 2024).

Figura 4

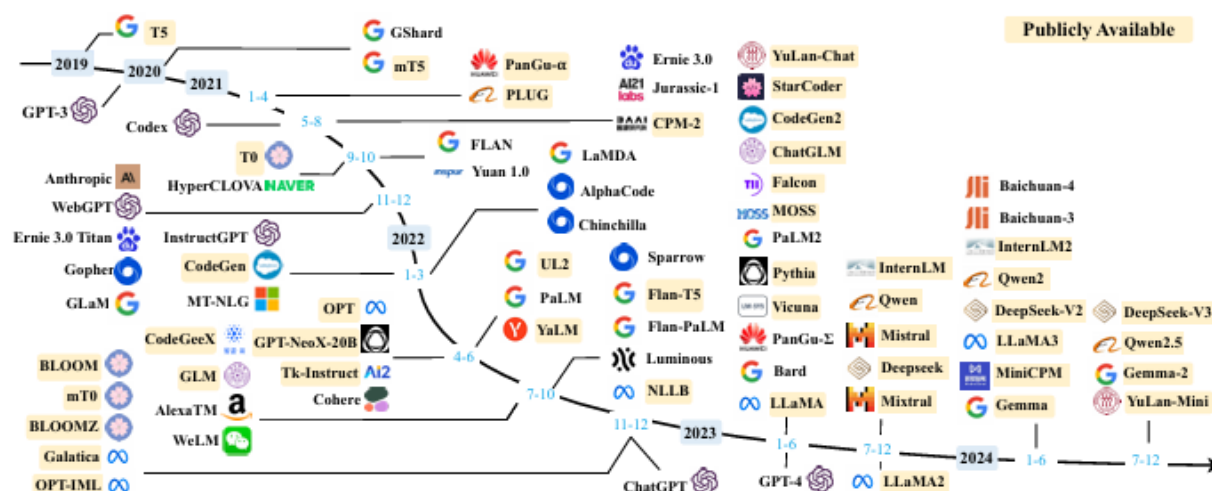
Evolución de “Language Models” desde el punto de vista de capacidad de resolución



(Zhao et al., 2023)

Figura 5

Timeline de “Large Language Models” (Mas de 10 Billones de parámetros)



(Zhao et al., 2023)

2.6.2 Capacidades Semánticas y Heurísticas

Los LLMs destacan por su habilidad para operar en espacios semánticos vectoriales, donde conceptos, intenciones y contextos se representan como distribuciones de alta dimensionalidad. Esta representación permite realizar tareas de similitud semántica, abstracción conceptual y transferencia de conocimiento entre dominios, lo que los convierte en agentes ideales para funciones de mediación y síntesis (Singh et al., 2023).

Además, los LLMs pueden incorporar heurísticas de decisión, como las utilizadas en planificación iterativa, razonamiento causal o validación argumentativa. Estas capacidades han sido exploradas en enfoques como ReAct (Yao et al., 2022), donde el agente razona en lenguaje natural mientras ejecuta acciones observables, y MAD (“Multi-Agent Debate”), que permite deliberación colaborativa entre múltiples agentes LLM.

2.6.3 Limitaciones y Herramientas Complementarias

A pesar de sus capacidades avanzadas, los LLMs presentan limitaciones importantes en áreas como:

- Razonamiento cuantitativo exacto
- Acceso a datos en tiempo real
- Manipulación estructurada de información

Para superar estas restricciones, se integran herramientas externas que amplían su funcionalidad.

Estas herramientas se agrupan en tres categorías principales:

1. Acceso a datos externos: APIs, motores de búsqueda, bases de datos.
2. Procesamiento simbólico: cálculos matemáticos, lógica formal.
3. Interacción estructurada: generación de código, edición de documentos, visualización de diagramas.

2.7 Herramientas en Arquitecturas Agenticas

2.7.1 Tipología de herramientas

En el contexto de arquitecturas agenticas basadas en modelos de lenguaje, las herramientas representan extensiones funcionales que permiten a los agentes superar sus limitaciones inherentes. Estas herramientas no solo amplían el rango de acción de los agentes, sino que también mejoran su precisión, verificabilidad y adaptabilidad a tareas del mundo real (Wu et al., 2023).

Las herramientas utilizadas por agentes LLM pueden clasificarse en tres grandes categorías:

1. **Herramientas de acceso a datos externos:** permiten consultar APIs, motores de búsqueda, bases de datos en vivo o documentos institucionales.

2. **Herramientas de cálculo y procesamiento simbólico:** utilizadas para resolver operaciones matemáticas, evaluar expresiones lógicas o realizar transformaciones precisas de datos.
3. **Herramientas de interacción estructurada:** incluyen generadores de código, editores de documentos, visualizadores de diagramas y conectores con plataformas externas (Slack, Notion, Jira, entre otros).

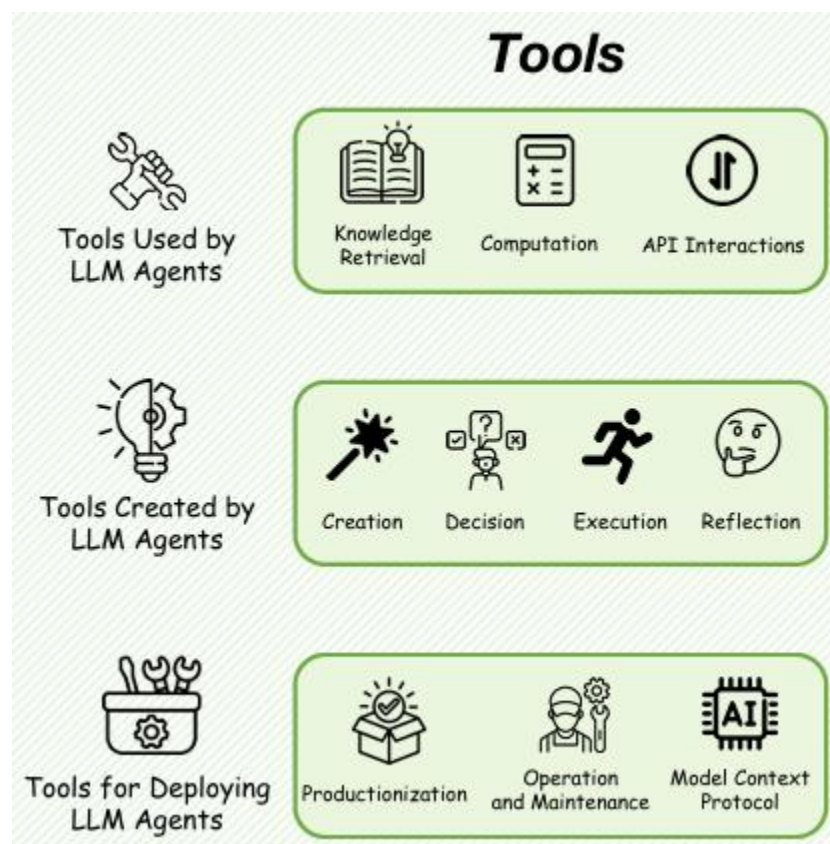
Esta tipología permite diseñar agentes modulares, donde cada componente puede invocar herramientas específicas según el tipo de tarea, manteniendo la coherencia semántica del sistema.

2.7.2 Autogeneración y Adaptabilidad

Una característica emergente en arquitecturas avanzadas como MetaGPT y ChatDev es la capacidad de los agentes para autogenerar herramientas. Esta funcionalidad se activa especialmente en tareas creativas o de resolución de problemas no estructurados, donde el agente puede construir funciones auxiliares, scripts o interfaces personalizadas para abordar desafíos específicos (Hong et al., 2024; Qin et al., 2023).

La autogeneración de herramientas no solo mejora la eficiencia operativa, sino que también permite una adaptabilidad dinámica al entorno, facilitando la evolución del sistema sin intervención humana directa. Esta propiedad es clave en entornos distribuidos, donde los requisitos cambian constantemente y los agentes deben responder de forma autónoma.

Figura 6
Tipos de herramientas



(Luo et al., 2025)

2.7.3 Protocolos MCP y Estandarización

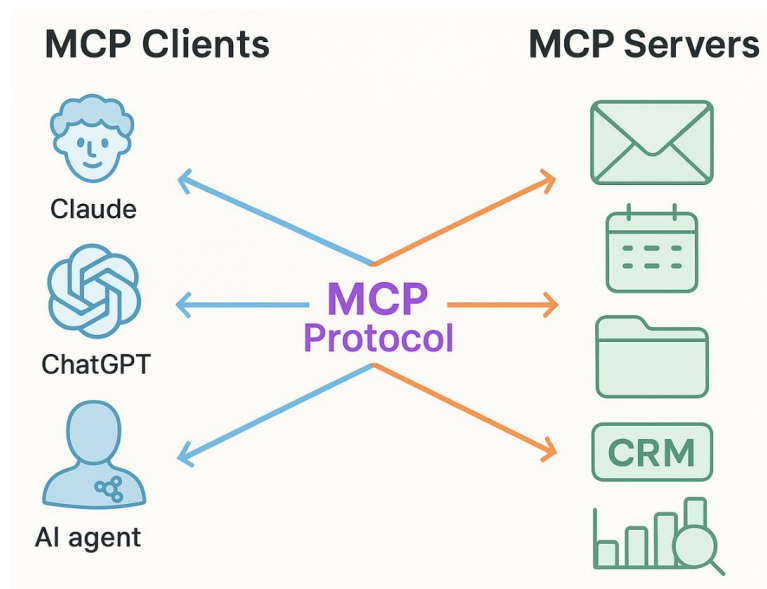
El desarrollo del “Model Context Protocol” (MCP) por parte de Anthropic ha marcado un hito en la estandarización de la interacción entre agentes LLM y herramientas externas. MCP propone una interfaz única, segura y modular que permite conectar modelos de lenguaje con múltiples fuentes de datos y funciones sin necesidad de reprogramar cada integración (Anthropic, 2024).

La arquitectura MCP se basa en un modelo cliente-servidor que incluye:

- **Hosts:** aplicaciones que requieren contexto (“IDEs”, “chatbots”).
- **Cientes MCP:** gestionan conexiones a servidores.
- **Servidores MCP:** exponen datos y herramientas accesibles al modelo.
- **Fuentes de datos:** locales (archivos, bases) o remotas (APIs).

Este protocolo ha sido adoptado por actores como OpenAI, DeepMind, Replit y Microsoft Copilot Studio, consolidándose como el “USB-C” de la inteligencia artificial conectada. En el marco TB-CSPN, MCP permite que los agentes consultores accedan a recursos externos de forma segura, trazable y escalable, fortaleciendo la coordinación semántica y la auditabilidad del sistema.

Figura 7
Protocolo MCP



(Gupta, 2025)

2.8 Enfoques Emergentes: ReAct y MAD

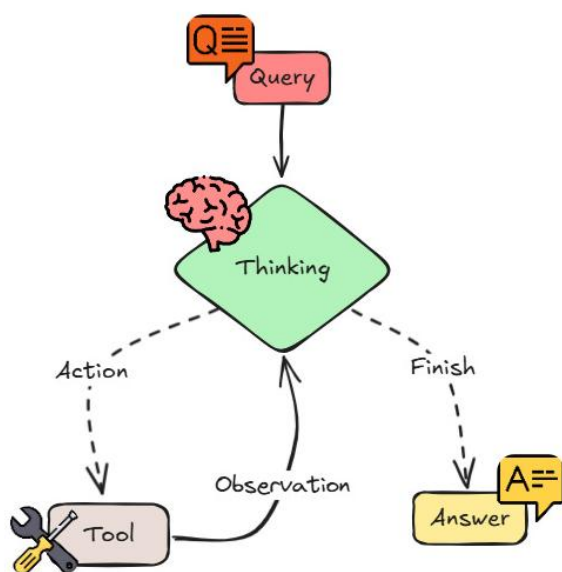
2.8.1 ReAct: Razonamiento Iterativo

El enfoque ReAct (“Reasoning and Acting”), introducido por Yao et al. (2022), propone una arquitectura híbrida en la que los agentes típicamente LLMs combinan razonamiento en lenguaje natural con ejecución de acciones observables. Esta estrategia permite que el agente formule hipótesis, actúe sobre el entorno, evalúe los resultados y ajuste su razonamiento en función de la retroalimentación obtenida.

En sistemas multiagente, ReAct representa una solución eficaz para entornos distribuidos con alta incertidumbre, donde la adaptabilidad y la interpretación contextual son esenciales. Al integrar ReAct en arquitecturas como TB-CSPN, los agentes consultores no solo generan conocimiento pasivamente, sino que participan activamente en la resolución de tareas, manteniendo la trazabilidad semántica de sus inferencias.

Además, el razonamiento intermedio se vuelve transparente para otros agentes, lo que facilita la coordinación entre capas funcionales (superficie, observación, cómputo) y refuerza la auditabilidad del sistema.

Figura 8
Arquitectura de un agente ReAct



(Andrés, 2024)

2.8.2 MAD: Deliberación Multiagente

El enfoque MAD (“Multi-Agent Debate”) se basa en la deliberación colaborativa entre múltiples agentes, generalmente LLMs, que discuten argumentos, contraargumentos y justificaciones para resolver problemas complejos o validar decisiones (Qin et al., 2023). Inspirado en principios de argumentación lógica y deliberación humana, MAD busca aprovechar la diversidad de perspectivas para alcanzar conclusiones más robustas y justificadas.

En el marco TB-CSPN, MAD puede integrarse como mecanismo de validación semántica en las capas de supervisión y observación. Por ejemplo:

- Agentes consultores debaten sobre la interpretación de requisitos ambiguos.
- Agentes especializados presentan estrategias alternativas de implementación.

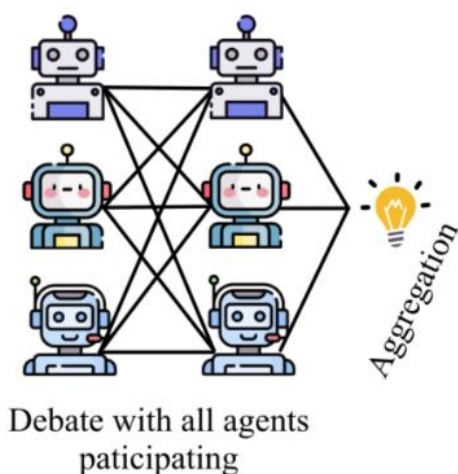
- Agentes centáuricos intervienen con criterios éticos o estratégicos.

Este enfoque permite detectar contradicciones, reducir sesgos y fortalecer la trazabilidad de decisiones, especialmente en contextos donde la ambigüedad o la subjetividad son elevadas.

Figura 9

Ejemplo de arquitectura MAD

Multi-agent Debate



(Cui et al., 2025)

2.8.3 Integración en TB-CSPN

La integración de ReAct y MAD en TB-CSPN aporta una capa adicional de inteligencia organizativa, al permitir que los agentes no solo ejecuten tareas, sino que también razonen colectivamente sobre ellas. Esta capacidad es clave para entornos híbridos humano-IA, donde la interpretación, la deliberación y la validación cruzada son fundamentales.

Ambos enfoques refuerzan la transparencia, la explicabilidad y la coherencia semántica del

sistema, alineándose con los principios de trazabilidad y supervisión jerárquica definidos en TB-CSPN. En conjunto, ReAct y MAD representan herramientas estratégicas para el diseño de agentes autónomos colaborativos, capaces de operar en entornos dinámicos y complejos con alto grado de responsabilidad semántica.

2.9 Modelado de Tópicos como Puente Semántico

2.9.1 LDA y Modelos Neuronales

El modelado de tópicos es una técnica fundamental para la mediación semántica entre agentes humanos e inteligencia artificial. A través de esta metodología, grandes volúmenes de texto pueden ser condensados en representaciones temáticas interpretables, facilitando la alineación conceptual entre distintos tipos de agentes (Blei et al., 2003).

Uno de los enfoques más representativos es el Latent Dirichlet Allocation (LDA), que modela los documentos como combinaciones probabilísticas de tópicos latentes, y los tópicos como distribuciones de palabras. Esta estructura permite identificar patrones semánticos recurrentes y organizar el conocimiento de forma interpretable.

En años recientes, se han desarrollado modelos de tópicos neuronales, que combinan la capacidad de representación profunda de las redes neuronales con la necesidad de mantener interpretabilidad. Estos modelos permiten una mayor flexibilidad en la detección de relaciones semánticas complejas, sin perder la trazabilidad conceptual que caracteriza al LDA.

2.9.2 Aplicaciones en Coordinación

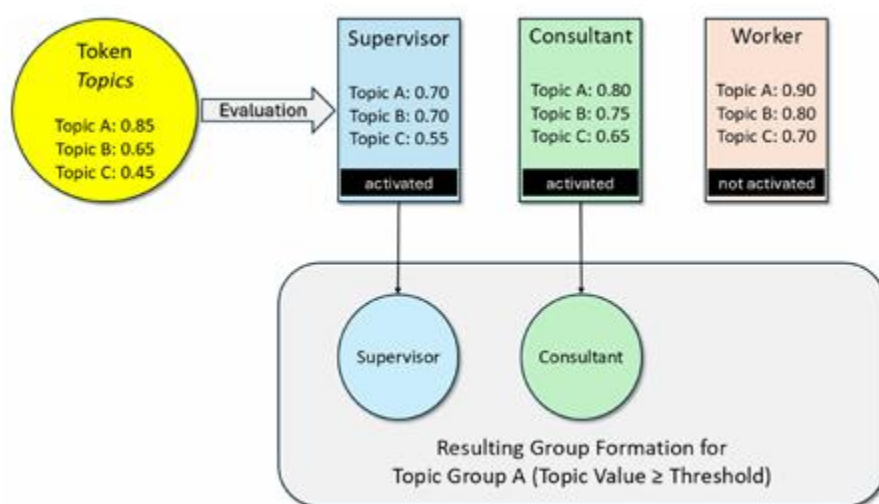
En el marco TB-CSPN, los tópicos actúan como vectores compartidos de significado y contexto, que regulan la activación de transiciones, la asignación de tareas y la formación dinámica de grupos de agentes. Esta función es especialmente relevante en sistemas híbridos, donde la semántica debe ser preservada a lo largo de múltiples capas funcionales.

Por ejemplo, los agentes consultores (LLMs) pueden enriquecer o recombinar tópicos para generar nuevas interpretaciones, mientras que los agentes ejecutores se suscriben dinámicamente a los tópicos más relevantes para filtrar la información necesaria. Los agentes supervisores, por su parte, pueden inspeccionar, modificar o validar los tópicos desde una perspectiva estratégica o ética.

Esta arquitectura temática permite una coordinación explícita entre agentes heterogéneos, superando las limitaciones de los sistemas opacos de aprendizaje profundo de extremo a extremo.

Figura 10

Ejemplo de suscripción de tópicos relevantes según el agente



(Borghoff et al, 2025)

2.9.3 Interpretabilidad y Control Semántico

Una de las principales ventajas del modelado de tópicos es su capacidad para introducir un nivel de alineación explícita entre la comprensión humana y la inferencia automática. A diferencia de los “embeddings” densos o las representaciones latentes no interpretables, los tópicos permiten establecer un puente semántico verificable entre agentes.

Esta propiedad es clave para garantizar la trazabilidad, la auditabilidad y el control semántico en sistemas críticos, como los que se diseñan bajo el marco TB-CSPN. Además, permite incorporar mecanismos de supervisión humana directa, sin necesidad de acceder a los procesos internos de inferencia de los modelos.

En suma, el modelado de tópicos constituye una herramienta estratégica para la construcción de sistemas multiagente semánticamente alineados, donde la colaboración entre humanos e IA se basa en estructuras conceptuales compartidas y verificables.

2.10 Representaciones Externas y Memoria Organizacional

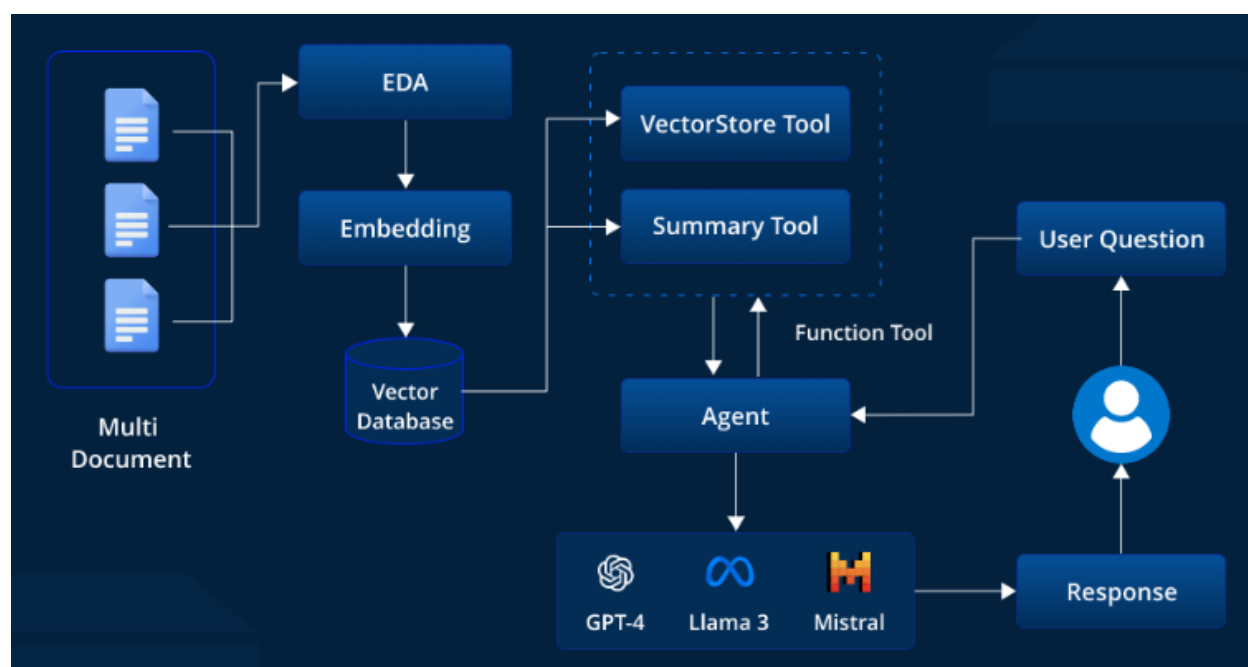
2.10.1 RAG y Recuperación Semántica

La incorporación de mecanismos de memoria organizacional compartida es esencial para garantizar la continuidad, trazabilidad y coherencia en sistemas multiagente. En el marco TB-CSPN, esta memoria no se limita al almacenamiento pasivo de datos, sino que se estructura como un repositorio semántico activo, accesible por todos los agentes del sistema.

Una tecnología clave en este contexto es el enfoque “Retrieval-Augmented Generation” (RAG), que combina la generación basada en modelos de lenguaje con la recuperación de información relevante desde una base de conocimientos. Este mecanismo permite que los agentes accedan a documentos históricos, especificaciones previas o registros de decisiones, enriqueciendo sus respuestas y asegurando la consistencia organizacional (Lewis et al., 2020).

RAG actúa como una memoria extendida, donde las representaciones semánticas se vinculan a fuentes verificables, facilitando la validación cruzada y la supervisión humana. En TB-CSPN, este enfoque potencia la capacidad de los agentes para operar en entornos dinámicos sin perder el contexto histórico ni la intención original.

Figura 11
Ejemplo de arquitectura de Agente y RAG



(LeewayHertz, 2024)

2.10.2 Diagramas, Logs y Trazabilidad

Además del texto, las representaciones externas estructuradas como diagramas, mapas de tareas, logs y planes desempeñan un papel fundamental en la coordinación entre agentes. Estas representaciones permiten:

- **Trazabilidad de decisiones:** cada acción puede vincularse a su origen semántico.
- **Incorporación de nuevos agentes:** facilita la integración en tareas en curso.
- **Validación por humanos supervisores:** mejora la auditabilidad del sistema.

- **Desambiguación en tareas complejas:** especialmente útil en contextos críticos.

El uso de formatos estructurados mejora la robustez semántica frente a la ambigüedad del lenguaje natural, permitiendo una comunicación más precisa entre agentes con arquitecturas distintas.

2.10.3 Memoria Persistente en Agentes Cognitivos

La incorporación de mecanismos de memoria persistente en agentes cognitivos representa un avance significativo respecto a arquitecturas basadas exclusivamente en recuperación externa como el RAG (“Retrieval-Augmented Generation”). Mientras que RAG se centra en enriquecer la generación de respuestas mediante la consulta de bases de conocimiento externas, las memorias persistentes permiten a los agentes almacenar, recuperar y continuar sesiones previas, manteniendo coherencia contextual y continuidad en el razonamiento.

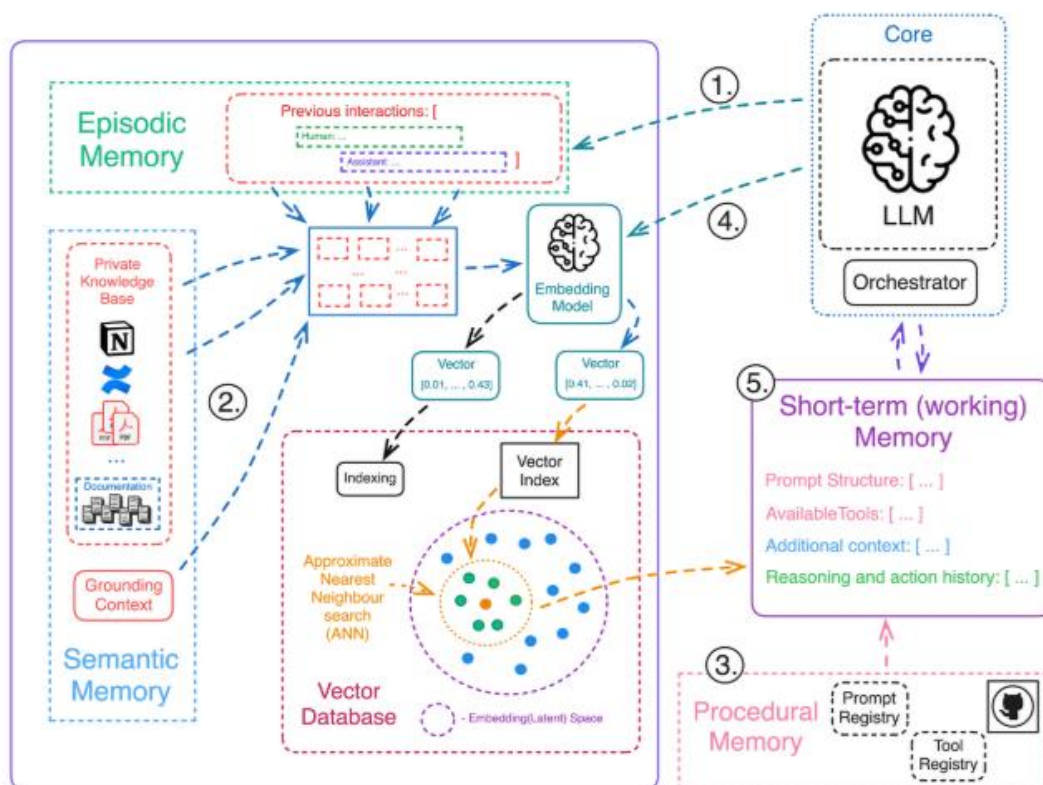
- **Memoria episódica**

La memoria episódica permite a los agentes registrar experiencias pasadas en forma de episodios estructurados, facilitando el aprendizaje adaptativo y la toma de decisiones basada en contexto. Esta arquitectura ha sido explorada en entornos multiagente donde la persistencia del estado es clave para la colaboración y la evolución del comportamiento.

- **Bases vectoriales persistentes**

Las bases vectoriales permiten almacenar representaciones semánticas de interacciones pasadas, facilitando la recuperación eficiente de contexto relevante. Cuando estas bases se diseñan para persistencia, los agentes pueden acceder a información histórica sin necesidad de reentrenamiento o reinicialización.

Figura 12
Tipos de memorias en Agentes



(Griciūnas, 2024)

2.11 Conclusión del Marco Teórico

El análisis teórico desarrollado en este capítulo ha permitido establecer los fundamentos conceptuales, arquitectónicos y metodológicos que sustentan el diseño e implementación de BusinessAnalystGPT. La evolución de los sistemas multiagente (MAS), la emergencia de la inteligencia artificial agéntica y la consolidación de arquitecturas híbridas humano-IA como TB-CSPN configuran un marco robusto para abordar los desafíos contemporáneos en la automatización de procesos cognitivos complejos, como la captura de requisitos funcionales.

BusinessAnalystGPT se inscribe en esta convergencia disciplinar al integrar agentes consultores basados en modelos de lenguaje de gran tamaño (LLMs), agentes ejecutores especializados y supervisión humana centáurica, todo ello articulado mediante protocolos normativos, ontologías compartidas y mecanismos de trazabilidad semántica. La incorporación de enfoques como ReAct y MAD refuerza la capacidad deliberativa y adaptativa del sistema, mientras que el uso de herramientas como RAG y memorias persistentes garantiza continuidad contextual y auditabilidad.

En este sentido, el marco teórico no solo proporciona una base conceptual para el proyecto, sino que también orienta sus decisiones de diseño, sus criterios de evaluación y su potencial de escalabilidad. BusinessAnalystGPT no es simplemente una aplicación técnica, sino una instancia experimental de los principios organizativos, semánticos y colaborativos que definen la nueva generación de sistemas inteligentes distribuidos. Su desarrollo representa, por tanto, una contribución práctica al estudio de la IA agéntica aplicada, alineada con los retos metodológicos y éticos que plantea la transformación digital acelerada.

3. Metodología

Este capítulo describe el enfoque metodológico empleado para evaluar el impacto de un asistente de IA en la eficiencia de la recopilación de requisitos en proyectos de software. En consonancia con las recomendaciones de investigación aplicada en sistemas multiagente y IA agentica, el estudio articula objetivos, hipótesis, diseño, instrumentos y procedimiento, y se apoya en el marco “Build–Collaborate–Evolve” como guía de implementación y evaluación iterativa.

Objetivos e hipótesis

Objetivo general: Determinar si la incorporación de un asistente de IA mejora la eficiencia del proceso de elicitation de requisitos frente a prácticas tradicionales.

Hipótesis de trabajo.

- **H1 (eficiencia):** El uso del asistente de IA reduce el tiempo total de recopilación y consolidación de requisitos.
- **H2 (calidad):** El uso del asistente de IA incrementa la completitud y coherencia de los documentos de especificación funcional.
- **H3 (satisfacción):** El uso del asistente de IA mejora la satisfacción percibida por Analistas y partes interesadas durante el proceso.

Operacionalizaciones

- **Eficiencia:** minutos totales hasta una FSD (“Functional Specification Document”) aceptada.
- **Calidad:** puntuación de completitud/consistencia mediante rúbrica experta.
- **Satisfacción:** cuestionario tipo Likert (utilidad, claridad, carga de trabajo).

Diseño

Se adopta un diseño experimental con comparación entre el proceso asistido por IA y el procedimiento de referencia. El enfoque incluye una prueba piloto para calibración y, posteriormente, una fase de validación en entorno controlado. El marco “Build–Collaborate–Evolve” orienta la construcción del agente, su colaboración orquestada con otros componentes y la evolución por retroalimentación.

Instrumentos

Cuestionarios de satisfacción (Likert) para utilidad percibida, claridad y carga de trabajo.

Métricas de desempeño (tiempos de respuesta, duración por fase, iteraciones hasta aceptación).

Rúbrica de calidad del entregable (completitud, coherencia, trazabilidad).

Entrevistas estructuradas para triangulación cualitativa y matices de adopción.

Procedimiento

El estudio se desarrolla en cuatro fases: diseño (definición de roles, variables e instrumentos), desarrollo (implementación del agente y flujos), prueba piloto (ajuste de parámetros y de instrumentos) y validación (recogida de datos y análisis). Los resultados alimentan ciclos de mejora conforme al enfoque “Build–Collaborate–Evolve”, preservando trazabilidad y auditabilidad del proceso.

En las secciones siguientes (3.1–3.6) se detallan el enfoque metodológico general, la construcción del agente, la colaboración multiagente, la evolución del sistema, los criterios de evaluación y el proceso experimental e implementación.

3.1 Enfoque Metodológico General

El diseño metodológico de este trabajo adopta el enfoque propuesto por el marco "Build - Collaborate - Evolve" (Luo et al., 2025), identificado como una de las estructuras más completas para analizar sistemas de agentes basados en LLMs. Esta metodología permite descomponer el ciclo de vida de un agente en tres dimensiones fundamentales: construcción, colaboración y evolución, abarcando desde el diseño de roles y planificación hasta la evaluación y mejora continua del sistema.

3.2 Construcción del Agente

Durante la fase de construcción, se definieron perfiles funcionales para los distintos tipos de agentes: Consultores (basados en LLM), Especializados (orientados a tareas concretas) y Supervisores (humanos o centáuricos). Este proceso se estructuró en torno a cuatro pilares interdependientes, que guían el diseño completo del agente:

- **Definición del perfil:** identificación de las capacidades, restricciones y tipo de interacción esperado para cada agente, asignando roles específicos dentro del sistema.
- **Mecanismo de memoria:** integración de memoria episódica y a largo plazo, incorporando técnicas como RAG para recuperación semántica contextualizada.
- **Capacidad de planificación:** desarrollo de módulos que permiten descomponer tareas complejas en subtarear, estableciendo secuencias lógicas y dependencias.
- **Ejecución de acciones:** implementación de interfaces que permiten a los agentes llevar a cabo acciones concretas, ya sea a través de generación de texto, invocación de herramientas o interacción con usuarios.

Estos componentes se diseñaron para operar de manera coordinada y modular, permitiendo la extensión y reconfiguración del sistema en función de nuevos requerimientos o contextos de uso.

3.3 Colaboración Multiagente

Para facilitar la colaboración entre agentes, se adoptó una arquitectura semánticamente mediada y orientada a tópicos, inspirada en el marco TB-CSPN (“Topic-Based Conditional Structured Process Networks”). Este enfoque permite organizar la interacción entre agentes según capas funcionales (superficie, observación, cómputo) y distribuir las transiciones de tareas en función del contexto temático, manteniendo así la trazabilidad y la alineación semántica entre intenciones humanas y acciones automáticas.

Cada agente, ya sea humano, consultor basado en LLM o especializado, actúa sobre transiciones específicas del sistema, suscribiéndose dinámicamente a tópicos relevantes. Esta estructura favorece una orquestación distribuida, donde los agentes colaboran sin perder su autonomía funcional ni su foco operativo.

Además, se incorporaron elementos del enfoque MAD (“Multi-Agent Debate”) como mecanismo de validación semántica. Los agentes consultores pueden participar en debates estructurados, donde presentan argumentos y contraargumentos para evaluar propuestas o decisiones generadas. Un agente supervisor o moderador humano puede intervenir para seleccionar la solución más coherente, promoviendo así transparencia, explicabilidad y mejora colectiva del sistema.

La combinación de TB-CSPN y MAD permite una colaboración robusta y dinámica, integrando razonamiento, discusión y ejecución de tareas dentro de un entorno controlado, verificable y alineado con los principios de diseño de sistemas multiagente avanzados.

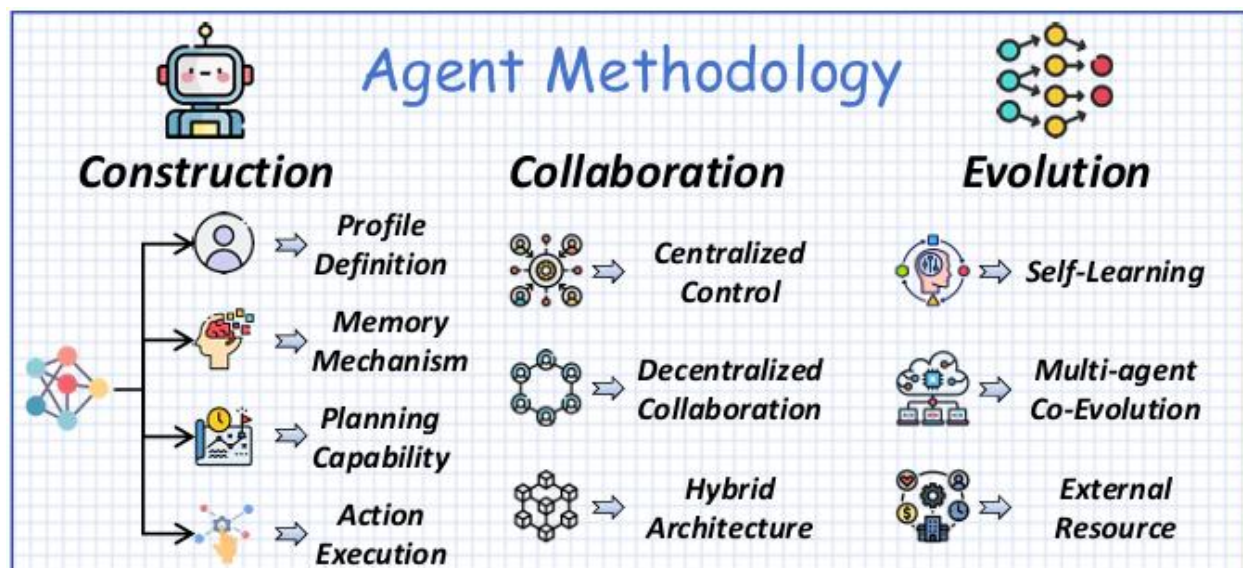
3.4 Evolución del Sistema

La capacidad evolutiva del sistema se implementó mediante ciclos de retroalimentación continua, donde los agentes son evaluados y ajustados a partir de los resultados de sus propias decisiones. Se emplearon agentes evaluadores encargados de analizar la calidad de las respuestas generadas, proponiendo ajustes con base en métricas internas de desempeño como coherencia semántica, completitud de requisitos y alineación con la intención del usuario.

Esta dinámica se complementa con técnicas de “trial-and-error”, “self-debugging” y “external feedback loops”, permitiendo una evolución adaptativa tanto desde el punto de vista técnico como semántico. Adicionalmente, el sistema integra memorias episódicas que registran el historial de interacción de cada sesión (conversaciones, documentos generados y diagramas de procesos). Cuando estas memorias son recuperadas, permiten no solo retomar el estado previo de la especificación funcional, sino también consolidar aprendizajes derivados de la retroalimentación del usuario.

En particular, las experiencias confirmadas como positivas son incorporadas al RAG principal, generando un mecanismo de aprendizaje acumulativo. De esta manera, cada nueva sesión no parte desde cero, sino que se beneficia de la evolución de interacciones anteriores, lo que contribuye a perfeccionar la calidad de las respuestas, optimizar la pertinencia de los documentos generados y reforzar la alineación con las expectativas del usuario.

El objetivo final es garantizar que el sistema se mantenga flexible, autorregulado y progresivamente más competente, consolidando un proceso de mejora continua que incrementa tanto su robustez técnica como su adecuación a contextos cada vez más complejos.

Figura 13*Metodología: Construir – Colaborar – Evolucionar**(Luo et al., 2025)*

3.5 Evaluación

La evaluación del sistema se llevó a cabo mediante herramientas propias desarrolladas para registro (“logging”), trazabilidad (“tracing”) y observabilidad, permitiendo monitorear el comportamiento del agente en cada etapa del flujo de interacción. Estas herramientas fueron diseñadas específicamente para capturar eventos clave, tiempos de respuesta, transiciones entre estados y decisiones tomadas por los agentes en cada capa del sistema TB-CSPN.

Además de los datos cuantitativos obtenidos a través de métricas internas, se aplicaron cuestionarios de satisfacción y entrevistas estructuradas a Analistas de negocio, con el fin de evaluar la utilidad percibida, la coherencia de los resultados y la facilidad de interpretación de las salidas generadas por el agente.

Si bien en esta primera fase no se incorporaron marcos de evaluación avanzados como AutoGenBench, se contempla su utilización en futuras iteraciones del proyecto, con el objetivo de comparar el rendimiento del sistema frente a” benchmarks” establecidos en tareas de planificación, razonamiento y colaboración multiagente.

3.6 Proceso Experimental e Implementación

El desarrollo del agente se realizó en cuatro fases:

3.6.1 Diseño Conceptual del Sistema y Definición de Roles de Agentes.

El flujo de trabajo se gestiona mediante un mecanismo de selección dinámica de agentes, que permite activar en cada momento a los participantes más adecuados según el contexto de la conversación y el estado del proceso. A continuación, se describen los agentes implementados:

- **planning_agent:** Es el primer agente en activarse. Su función principal es descomponer la tarea global en subtareas manejables y coordinar la ejecución secuencial del equipo. Para llevar a cabo esta coordinación, el agente analiza el contenido del chat compartido entre todos los agentes, identifica el tópico predominante o el estado actual de la conversación, y decide el siguiente paso lógico a realizar. Publica esta decisión en el chat compartido, lo que permite que el sistema “SelectorGroupChat” utilice dicho tópico como criterio para seleccionar al próximo agente que debe intervenir.
- **interviewer_agent:** actúa como Analista de negocio; entrevista al usuario para recopilar los requisitos del proyecto. Utiliza memoria vectorial RAG para adaptar las preguntas al contexto y alimentar una plantilla de especificación funcional.
- **summarizer_agent:** sintetiza la información obtenida en la entrevista. Entrega un resumen al usuario y valida si desea ampliar la información. En caso afirmativo, reactiva al interviewer_agent.

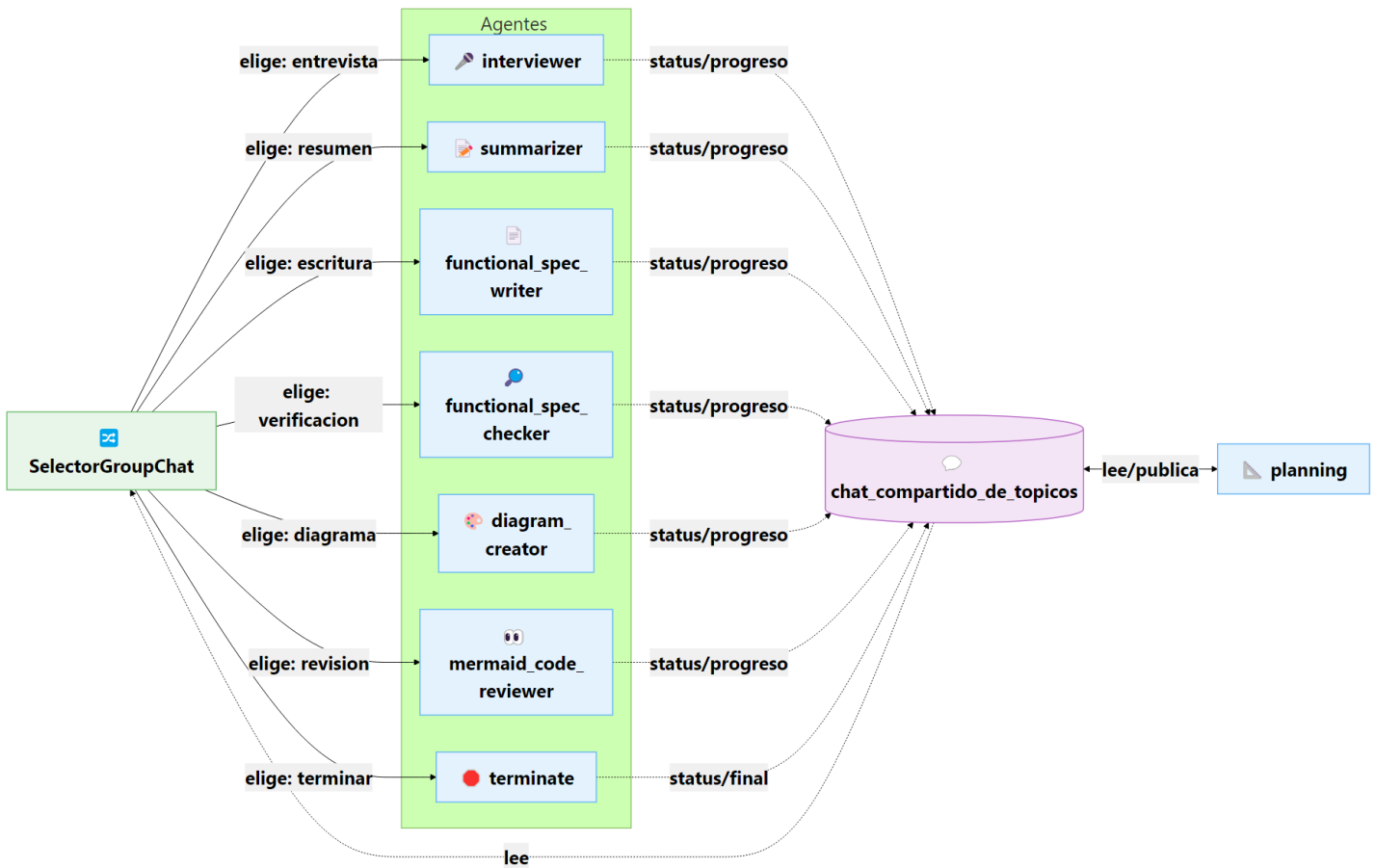
- **functional_spec_writer_agent:** utiliza el resumen generado para redactar un documento de especificación funcional en formato MD, siguiendo una estructura estandarizada y exhaustiva.
- **functional_spec_checker_agent:** revisa si el documento generado está completo. Evalúa la cobertura de todas las secciones obligatorias y señala cualquier omisión para su corrección.
- **use_case_agent:** genera casos de uso e historias de usuario a partir de la información recolectada del proyecto.
- **glossary_agent:** construye un glosario de términos a partir de los requisitos recopilados y la información del proyecto.
- **diagram_creator_agent:** genera un diagrama del proceso descrito en el documento funcional, utilizando el lenguaje Mermaid. Escribe el código generado en un archivo.md en canvas.
- **mermaid_code_reviewer_agent:** revisa el código generado, identifica posibles mejoras o faltantes, y valida su completitud. Si no hay observaciones, genera la versión final en.mmd.
- **user_proxy:** representa la interacción humana dentro del sistema, permitiendo que el flujo conversacional se mantenga controlado y rastreable.
- **terminate_agent:** Este agente se activa en la fase final del proceso. Su responsabilidad principal es ejecutar las tareas de cierre, incluyendo la generación del PDF final que contiene las especificaciones de los requerimientos, los diagramas generados y cualquier otro artefacto relevante. Además, crea el directorio de experiencia, donde se organiza y almacena toda la información útil de la sesión. También guarda los datos clave en la

memoria episódica, lo que permite recuperar el contexto y el historial de decisiones en futuras ejecuciones o análisis retrospectivos.

- **load_experience_agent:** habilita la reapertura y continuación de una experiencia previa identificada por `session_id`. Su objetivo es restaurar el estado operativo mínimo (artefactos y contexto) para retomar el flujo estándar de orquestación (`planning_agent` → `SelectorGroupChat` → resto de agentes) sin pérdida de trazabilidad y así agregar “feedback” y mejoras a la especificación funcional existente.
- **Load_conversation_script_agent:** ingiere un guion/”transcript” de conversación externa (p. ej., Teams/Zoom/Meet/llamada Realtime), limpiarlo y normalizarlo (segmentación por turnos, mapeo de hablantes, detección de idioma, deduplicaciones) y preparar un paquete textual único para que el `planning_agent` orqueste el envío directo al `summarizer_agent`.

Figura 14

Ejemplo de comunicación entre agentes para el caso con entrevista por primera vez



“System Prompt” de los agentes (ver en Apéndice D)

Este diseño, alineado con TB-CSPN y enriquecido por dinámicas de deliberación inspiradas en MAD, garantiza una coordinación semántica efectiva, modularidad organizativa y trazabilidad completa de la información generada por los agentes en cada fase del proceso.

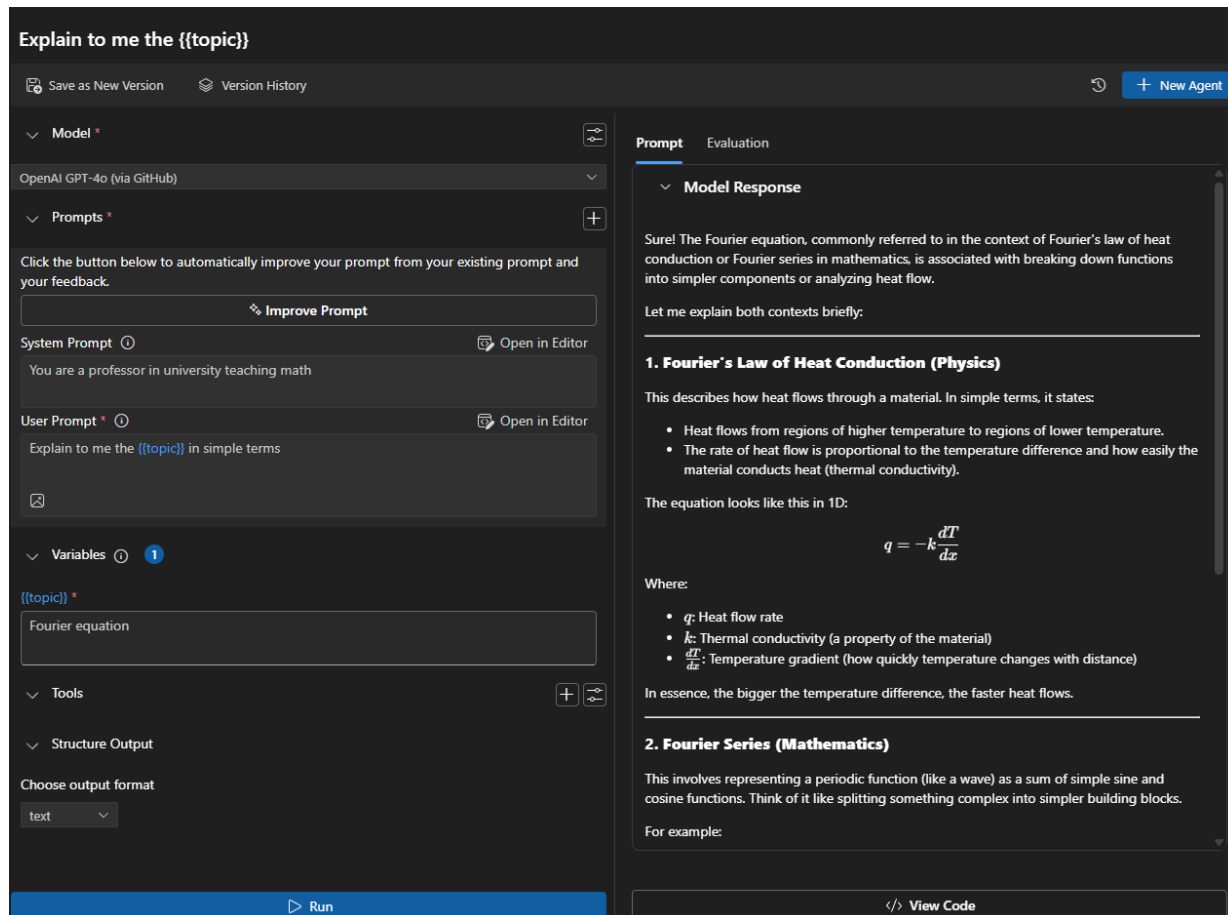
También cabe mencionar que, durante la fase de definición de roles y configuración de los agentes, uno de los mayores desafíos técnicos fue el diseño y ajuste de los “system messages” de cada agente. Estos mensajes definen el comportamiento, límites, estilo de respuesta y objetivos operativos de cada entidad, por lo que su correcta formulación es clave para la coherencia general del sistema.

Este proceso requirió una intensa labor de “prompt engineering” y una gran cantidad de pruebas manuales, ya que cada ajuste implicaba ejecutar el flujo completo de agentes para validar si el comportamiento resultante era el esperado. Las iteraciones fueron numerosas, especialmente cuando se utilizaban modelos locales tipo SLM (“Small Language Models”), cuya sensibilidad a la formulación del “prompt” es significativamente mayor en comparación con modelos LLM de última generación como GPT-4.

Este ciclo de prueba y error consumió una gran cantidad de tiempo durante las etapas iniciales del desarrollo. Recién en fases avanzadas del proyecto se identificó una herramienta que podría haber optimizado notablemente este trabajo: el plugin "AI Toolkit" de Microsoft para Visual Studio Code. Esta extensión permite diseñar y probar “prompts” de forma interactiva, con soporte tanto para modelos locales como en la nube, lo que facilita un entorno más ágil para experimentar con diferentes configuraciones antes de integrarlas en el flujo operativo completo.

Esta experiencia resalta la importancia de contar con herramientas específicas de desarrollo para entornos multiagente, donde el ajuste fino del comportamiento conversacional de cada componente es esencial para el éxito global del sistema. (Ver Apéndice D para entender un poco mas del trabajo hecho en “prompts”)

Figura 15
Screenshot de “MS AI TOOLKIT”



3.6.2 Implementación Técnica del Agente BusinessAnalystGPT, Utilizando una Arquitectura Modular.

La implementación técnica del sistema BusinessAnalystGPT se estructuró sobre una arquitectura multiagente modular, seleccionada tras una evaluación comparativa de plataformas líderes en el desarrollo y despliegue de agentes basados en modelos de lenguaje (LLMs). Se analizaron propuestas como LangChain, LlamaIndex, MetaGPT, ChatDev y AutoGen, cada una con distintos niveles de madurez, flexibilidad y alineación con paradigmas organizativos avanzados.

Además de considerar la arquitectura y filosofía de coordinación de cada “framework”, se evaluaron también sus capacidades desde el punto de vista del despliegue, operación y mantenimiento, según el rol que desempeñan en el ciclo de vida de un agente LLM:

Herramientas de producción: AutoGen se distingue como una plataforma “open-source” extensible y personalizable que permite a los desarrolladores construir agentes conversacionales multiagente, adaptando flujos de trabajo y módulos a necesidades específicas. Por su parte, LlamaIndex ofrece herramientas orientadas al acceso, indexación y recuperación de datos locales, actuando como una infraestructura centrada en la gestión semántica de datos para aplicaciones LLM. LangChain, en particular, es un “framework” “open-source” ampliamente adoptado que permite construir aplicaciones basadas en LLMs con un alto grado de extensibilidad y personalización, facilitando la creación de módulos y flujos de trabajo adaptados a necesidades específicas. Esta flexibilidad ha hecho de LangChain una herramienta popular en prototipado rápido y orquestación de componentes modulares en aplicaciones de IA generativa. MetaGPT simula un entorno industrial clásico, donde cada fase del flujo está asignada a un agente con un rol funcional fijo (p. ej., “project manager”, “developer”, “QA”), controlado por un planificador central. Este diseño, aunque eficaz en contextos deterministas, restringe la autonomía local de los agentes y su capacidad para adaptarse a entornos cambiantes. MetaGPT forma parte de una línea de investigación de sistemas con controlador explícito, junto con modelos como *The Coscientist* o *LLM-Blender*, donde la toma de decisiones está centralizada o automatizada por un ente superior que controla el flujo completo.

Paradigmas comparados: ReAct vs. TB-CSPN

El análisis reveló dos enfoques metodológicos claramente diferenciados:

- Paradigma ReAct / Reflexión iterativa, como el que implementan MetaGPT, ChatDev, LangChain donde los agentes ejecutan acciones, reflexionan sobre ellas, y ajustan su razonamiento en ciclos secuenciales. Estos sistemas operan con roles estáticos y protocolos deterministas, útiles para tareas estructuradas como desarrollo de software o tareas académicas.
- Paradigma temático y deliberativo (TB-CSPN), representado por AutoGen y OpenAgents, donde los agentes colaboran mediante tópicos compartidos, roles dinámicos y capas de interacción (superficie, observación, ejecución). Este enfoque facilita la adaptación semántica, la trazabilidad y la orquestación distribuida en entornos con alta incertidumbre.

Autogen como sistema deliberativo y adaptable

AutoGen, en cambio, representa un sistema basado en comunicación estructurada e interacción deliberativa, más cercano a las necesidades del presente proyecto. Permite la creación de agentes con roles definidos pero reconfigurables, que colaboran mediante controladores de flujo que intercambian argumentos y refinan decisiones colectivamente. Esta arquitectura no sólo está alineada con TB-CSPN, sino que también integra principios de MAD (“Multi-Agent Debate”) y coevolución competitiva.

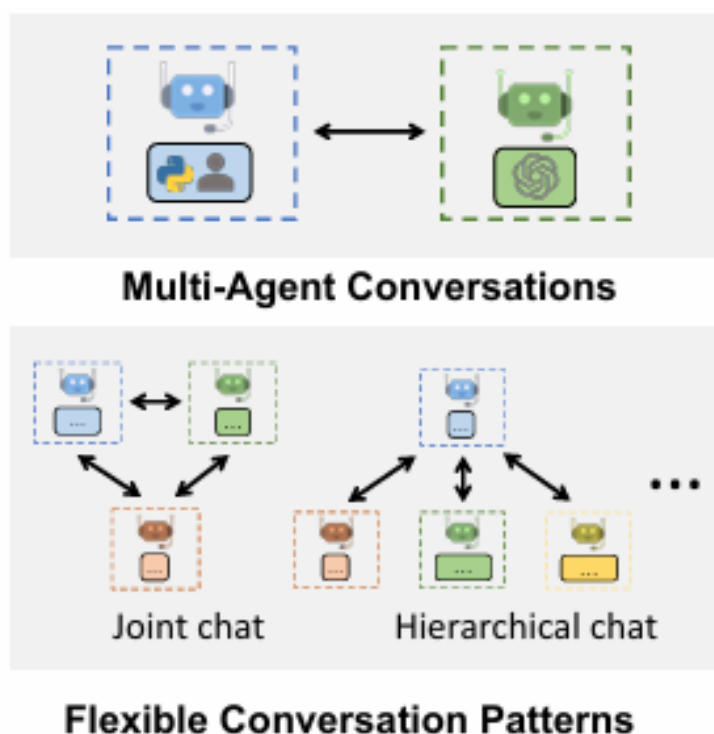
AutoGen permite la participación de múltiples agentes en debates iterativos estructurados, en los que se contrastan ideas, se corrigen errores, y se construyen soluciones más sólidas y verificables. Esta dinámica ha sido reconocida como fundamental en trabajos recientes como MADR y MDebate, donde la confrontación estratégica entre agentes estimula el pensamiento divergente, mejora la alineación semántica y reduce problemas como la alucinación de LLMs o la convergencia prematura.

Además, el soporte de AutoGen para coevolución competitiva y adversarial, por ejemplo, mediante simulaciones de “red-teaming” entre agentes LLM refuerza su capacidad para generar agentes más resilientes, adaptativos y estratégicamente inteligentes. Esta propiedad es especialmente relevante para sistemas como BusinessAnalystGPT, que deben operar de forma segura, trazable y alineada con intenciones humanas explícitas.

Otro punto distintivo es que AutoGen ofrece soporte nativo para el protocolo MCP (“Model Context Protocol”), lo que permite a los agentes integrarse directamente con fuentes de datos externas, herramientas especializadas y flujos de trabajo empresariales mediante enlaces seguros y estandarizados.

Figura 16

Ejemplos de diferentes arquitecturas con AutoGen



(Wu et al., 2023)

AutoGen como controlador de flujo

En sistemas multiagente modernos, la capacidad de coordinar agentes sin depender de un orquestador central rígido es clave para lograr escalabilidad, flexibilidad y adaptabilidad semántica. En este contexto, AutoGen incorpora varios mecanismos avanzados de control de flujo conversacional, entre los que destacan SelectorGroupChat y el recientemente adoptado patrón Swarm el cual está inspirados por el diseño de OpenAgents, un sistema introducido por OpenAI.

SelectorGroupChat: selección basada en modelos generativos

SelectorGroupChat es una arquitectura de chat grupal que permite a múltiples agentes participar en una conversación estructurada. A diferencia de esquemas secuenciales como RoundRobin, en SelectorGroupChat, el siguiente agente en hablar se selecciona dinámicamente usando un modelo generativo (por ejemplo, un LLM). Este modelo analiza el contexto compartido (historial de conversación, roles y descripciones de los agentes) y determina cuál agente está mejor posicionado para intervenir a continuación.

Las características clave de SelectorGroupChat incluyen:

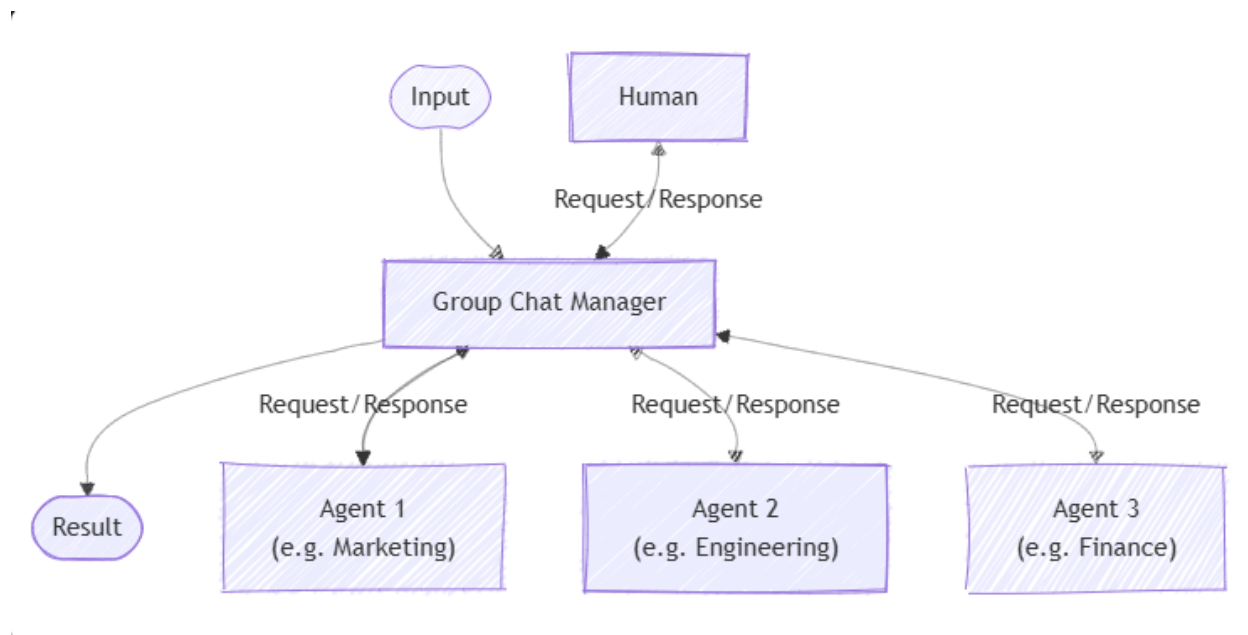
- Selección del hablante basada en modelos (“model-based speaker selection”).
- Configuración de roles y descripciones de cada agente participante.
- Prevención de turnos repetidos por el mismo agente (a menos que se permita explícitamente).
- Funciones de selección personalizables, que permiten definir cómo se elige el próximo hablante o restringir el conjunto de candidatos.

Este enfoque favorece una colaboración dinámica y sensible al contexto, en la que los agentes actúan de forma autónoma pero coherente con el objetivo general de la conversación.

Todo el historial conversacional se mantiene como contexto compartido, lo que facilita la trazabilidad y permite que nuevas tareas se conecten con sesiones anteriores.

Figura 17

Ejemplo de arquitectura GroupChat



(Chen & Rickman, 2025)

Swarm: delegación distribuida con control local

Swarm representa un paso más allá en la descentralización del control, al implementar un patrón en el que los agentes pueden delegar tareas entre sí de manera directa, sin que exista un coordinador central. Esta estrategia, introducida por OpenAI dentro del sistema Swarm de OpenAgents, simula dinámicas de colaboración similares a equipos humanos distribuidos.

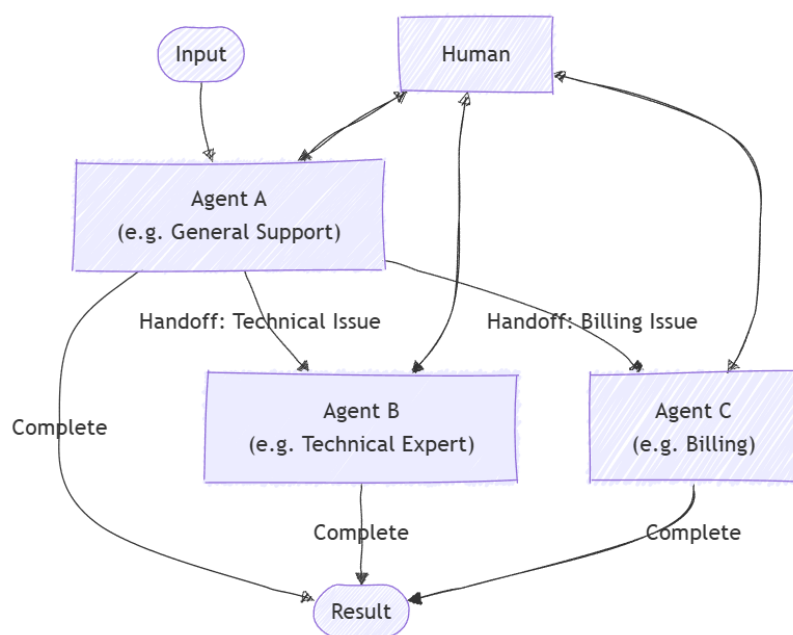
En Swarm:

- Cada agente tiene la capacidad de generar un “HandoffMessage”, señalando a qué otro agente transfiere el control de la tarea.
- Todos los agentes comparten el mismo contexto de conversación, lo que permite que las decisiones locales tengan impacto global.
- El flujo se organiza en turnos, donde el agente hablante es seleccionado en función del último mensaje de traspaso (HandoffMessage).
- El sistema continúa hasta cumplir una condición de término explícita, como completar la tarea o alcanzar un límite de iteraciones.

Este patrón refleja un sistema donde la lógica de planificación está distribuida entre los agentes, promoviendo la autonomía, la colaboración directa y la adaptación en tiempo real, características fundamentales para entornos complejos y no deterministas.

Figura 18

Ejemplo arquitectura Swarm con handoff



(Chen & Rickman, 2025)

AutoGen como controlador conversacional.

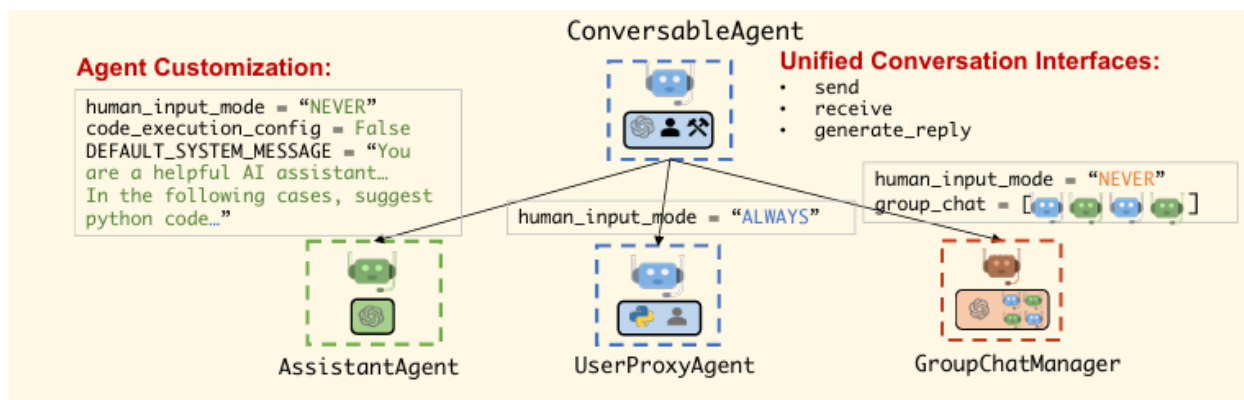
Dentro del sistema BusinessAnalystGPT, AutoGen no actúa como un controlador centralizado tradicional, sino como un entorno que habilita y media la interacción entre agentes autónomos, siguiendo una lógica más cercana a TB-CSPN que a arquitecturas de planificación secuencial.

Mediante SelectorGroupChat, AutoGen permite que los agentes respondan según su competencia contextual, mientras que con Swarm se habilita la delegación de tareas. Esta dualidad convierte a AutoGen en una plataforma híbrida, capaz de ejecutar procesos colaborativos que requieren:

- Coordinación semántica a través de tópicos y objetivos compartidos.
- Transferencia de control entre agentes en función del estado del diálogo.
- Colaboración contextualizada y controlada por modelos generativos.

Por tanto, AutoGen puede considerarse un controlador de flujo conversacional híbrido, en el que la gestión del proceso no se impone desde un nodo superior, sino que emerge desde las decisiones locales de cada agente dentro de un marco de reglas y objetivos compartidos. Esta propiedad lo convierte en la elección ideal para sistemas alineados con paradigmas como TB-CSPN y MAD, en los que la semántica, la trazabilidad y la deliberación entre agentes son componentes esenciales.

Figura 19
Ejemplo de Agentes AutoGen



(Wu et al., 2023)

3.6.3 Prueba piloto con casos de uso reales en entornos controlados

La validación del sistema se desarrolló bajo dos enfoques principales: una evaluación comparativa de “frameworks LLM”, y pruebas piloto con casos de uso representativos, utilizando distintas herramientas y entornos para garantizar funcionalidad, trazabilidad y adecuación al entorno de producción.

Evaluación práctica de “frameworks”

Para comparar “frameworks”, se llevaron a cabo pruebas utilizando LangChain, LlamaIndex y AutoGen, configurando “workflows” simples de “chatbot” que integraban RAG y un esquema de interacción “human-in-the-loop”. No se incluyen pruebas con ChatDev, MetaGPT u OpenAgents, ya que su enfoque no se alineaba con el objetivo exploratorio de esta etapa.

Principales resultados:

- LangChain demostró gran extensibilidad para crear módulos personalizados y el mayor soporte de integración con terceras partes.
- LlamaIndex destacó en manejo de datos locales y consultas semánticas.
- AutoGen resultó ser el “framework” más ágil y sencillo de configurar, con integración directa de agentes conversacionales, memoria contextual y soporte para flujos iterativos con SelectorGroupChat (“System Prompt” ver en Apéndice D) y Swarm.

Entornos de prueba y herramientas

Para las pruebas piloto se empleó un “Jupyter Notebook”, aprovechando su flexibilidad, capacidad interactiva para “*debugging*” y corrección en tiempo real. Simultáneamente se desarrolló una interfaz web usando Chainlit, simulando un entorno más cercano a producción, interactivo y representable.

En cuanto a los modelos LLM, se evaluaron tanto soluciones locales como en nube:

Modelos locales desplegados con Ollama y LM Studio: se probaron modelos de 8B, 12B, 20B+ y 32B parámetros (como Llama 3 y 4, DeepSeek R1, Qwen, Mistral, Gemma 3, Phi 4 y GPT-OSS). Aunque alcanzaron resultados aceptables, mostraron tendencia a confundirse o no completar tareas, debido a su naturaleza SLM y las restricciones de hardware. Por razones técnicas no fue posible probar versiones de 32B, las cuales, según “benchmarks”, se acercan en desempeño a los LLM más avanzados.

Actualización agosto 2025: La incorporación de nuevo hardware permitió la experimentación con modelos de hasta 32 mil millones de parámetros. Entre ellos, Gemma 3 demostró ser el más eficiente en cuanto a la capacidad de seguir instrucciones y completar tareas dentro de los plazos previstos. En contraste, los modelos especializados en “reasoning” no

resultaron adecuados para arquitecturas basadas en tópicos, dado que tienden a introducir ruido en el canal conversacional compartido mediante la inserción de trazas internas de razonamiento, comúnmente denominadas “Think”. Actualmente, el módulo SelectorGroupChat carece de la capacidad nativa para discriminar estas etiquetas, lo que con frecuencia conduce a confusiones contextuales. Si bien este problema puede mitigarse mediante funciones de selección personalizadas que filtren los contenidos asociados a las etiquetas “Think”, esta solución reduce la eficiencia general del sistema y mantiene un riesgo latente de ambigüedad en la interpretación de los mensajes.

En la nube, se utilizó exclusivamente OpenAI GPT-4o, con más de 100 billones de parámetros.

Este modelo ofreció:

- Respuestas casi instantáneas sin retardos perceptibles.
- Tasa de éxito superior al 99 % en tareas de especificación funcional, consultas RAG y “workflows” colaborativos.

En lo que respecta al RAG (“Retrieval-Augmented Generation”), se empleó ChromaDB como base de datos vectorial y *Sentence Transformers (all-MiniLM-L6-v2)* como modelo de “embeddings”. ChromaDB fue seleccionada por su facilidad de configuración y su elevado rendimiento en entornos locales, lo cual la convierte en una opción ampliamente utilizada en aplicaciones de recuperación de información aumentada. En este RAG se indexaron documentos en formato PDF con información metodológica sobre entrevistas para la recolección de requisitos, incluyendo tipos de preguntas y plantillas de documentos de especificaciones funcionales. Asimismo, se integraron páginas web relevantes sobre técnicas de entrevistas, y se habilitó un esquema de indexación automática: al incorporar nuevos documentos en la carpeta de

“*knowledge base (KB)*” o al especificar nuevas direcciones web, el sistema los indexa de manera inmediata sin necesidad de reentrenamiento.

Actualización agosto 2025: En cuanto a la memoria episódica y evolución por retroalimentación, también se utilizó ChromaDB en conjunto con *Sentence Transformers*. En este módulo se almacenaron el número de sesión, las conversaciones del usuario con el agente, las especificaciones funcionales generadas en formato Markdown y los diagramas de procesos en *Mermaid Markdown*. Este enfoque posibilita retomar sesiones previas para continuar mejorando los documentos mediante retroalimentación, además de formular preguntas específicas sobre documentos anteriores o interacciones pasadas. Una vez que el usuario confirma su satisfacción con el resultado final de la entrevista y el documento, la experiencia se transfiere al RAG principal, generando aprendizaje acumulativo. Cabe destacar que únicamente las memorias positivas son consolidadas en el repositorio principal, lo que asegura que el sistema evolucione sobre la base de experiencias validadas.

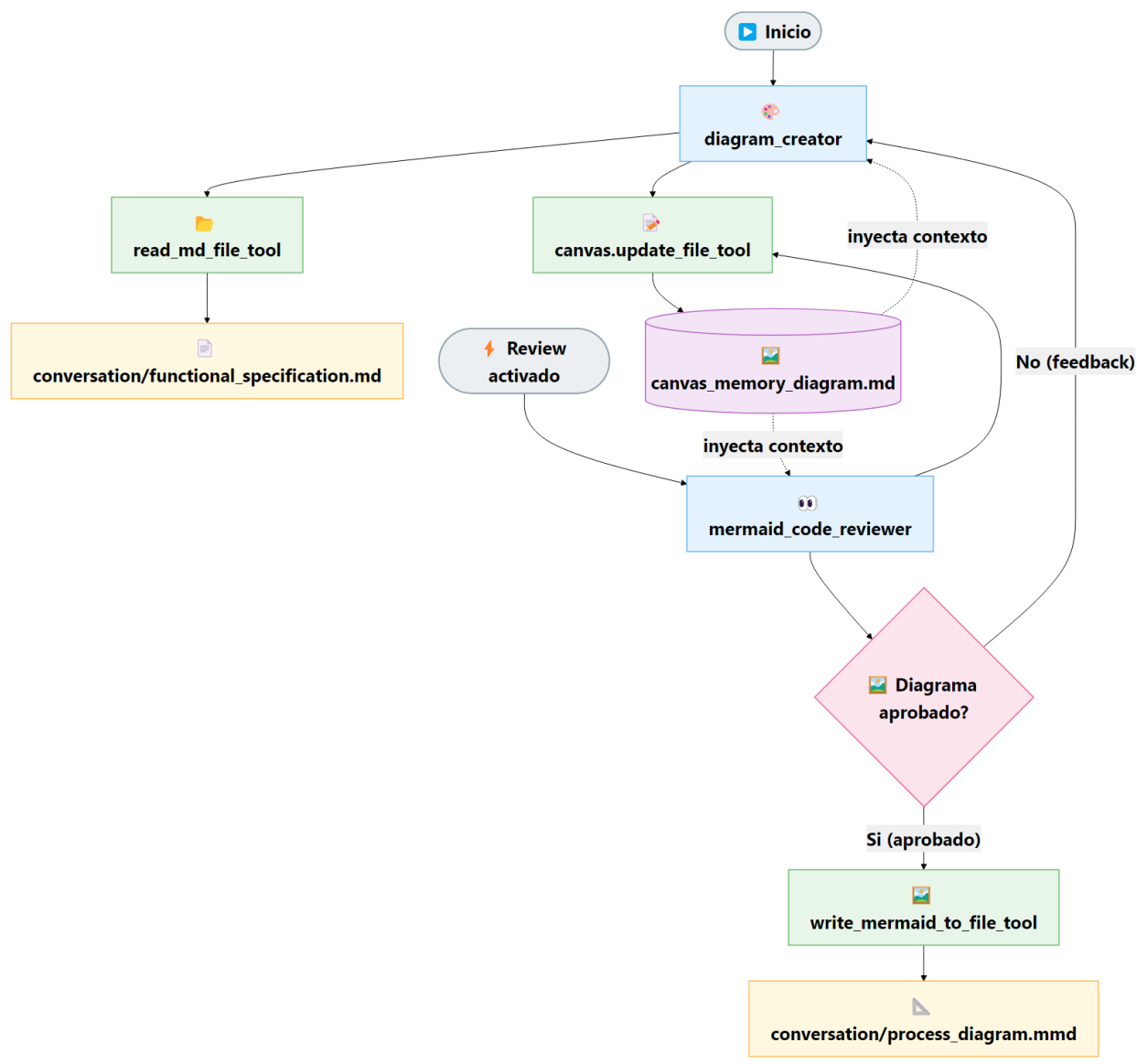
Además del RAG y la memoria episódica, se implementó un tercer mecanismo de almacenamiento denominado Canvas Memory. Este componente constituye un lienzo en memoria para archivos de texto y documentos en formato Markdown, conservando un historial completo de revisiones. Su núcleo, el módulo *TextCanvasMemory*, proporciona un sistema de almacenamiento persistente similar a un archivo, que puede ser leído y escrito por los agentes. Antes de cada inferencia, el sistema inyecta automáticamente el estado actualizado de los archivos en el contexto del modelo, lo que asegura la coherencia entre las interacciones.

Este tipo de memoria resulta particularmente útil para:

- Permitir que los agentes creen y modifiquen documentos de manera iterativa a lo largo de múltiples turnos.
- Habilitar la edición colaborativa de documentos entre varios agentes.
- Mantener un estado persistente a través de sucesivas rondas de conversación.
- Gestionar contenidos demasiado extensos para ser procesados en un único mensaje.
- El *canvas* incluye herramientas específicas que permiten:
- Crear o actualizar archivos con nuevo contenido.
- Aplicar parches en formato *unified diff* sobre archivos ya existentes.

En el contexto de este trabajo, la Canvas Memory se utilizó para que el *Diagram Creator Agent* y el *Mermaid Code Checker* colaboraran en tiempo real. Gracias a esta estrategia, ambos agentes pudieron validar que el código *Mermaid* fuera sintácticamente correcto y que los diagramas cubrieran de manera integral el proceso modelado, sin necesidad de escribir en disco ni sobrecargar el contexto de conversación.

Figura 20
Diagrama de flujo diagram_creator agent, mermaid_code_reviewer_agent y canvas_memory



Nota. Canvas Memory inyecta el estado actual de canvas_diagram.md al contexto del agente antes de cada inferencia

Herramientas de agente utilizadas:

El ecosistema de herramientas evolucionó en paralelo con las capacidades del marco AutoGen. Inicialmente, este “framework” solo soportaba FunctionTool, es decir, funciones en Python registradas automáticamente como herramientas de agente. En esta etapa inicial se diseñaron las siguientes funciones personalizadas:

- **agents_tools_helpers.write_md_file_tool:** utilizada por el Functional Spec Writer Agent para redactar la especificación funcional en formato Markdown, con posibilidad de añadir correcciones o actualizaciones.
- **agents_tools_helpers.read_md_file_tool:** empleada por el Functional Spec Checker Agent para leer la especificación funcional y validar su completitud.
- **canvas.update_file_tool:** usada por el Diagram Creator Agent y el Mermaid Code Checker, permitiendo trabajar directamente sobre una canvas memory sin necesidad de acceder al sistema de archivos, lo que favorece la colaboración en tiempo real para documentos pequeños.

Posteriormente, AutoGen incorporó soporte para MCP (“Model Context Protocol”), lo que permitió ampliar las herramientas a servidores MCP ya existentes y otros desarrollados ad hoc mediante FastMCP. En este nivel se implementaron:

- **PyMCP_FS_mcp_tools:** servidor existente que permite a los agentes realizar operaciones seguras sobre archivos y carpetas (lectura, escritura, copia, movimiento y borrado).
- **mermaid_mcp_tools:** servidor propio creado, que posibilita renderizar diagramas en Mermaid Code a formatos JPG, PNG y SVG, con el fin de integrarlos en los documentos PDF finales.

- **pdf_format_mcp_tools:** servidor propio creado, que formatea adecuadamente archivos Markdown para su conversión a PDF, incorporando automáticamente las imágenes de diagramas renderizados.

Finalmente, la última innovación —no disponible aún en otros frameworks— fue la utilización de agentes como herramientas (“Agent Tool”). Este enfoque resultó especialmente útil para delegar subtarefas especializadas sin sobrecargar a los agentes principales. Se destacan dos implementaciones:

- **use_case_agent:** empleado por el Functional Spec Writer Agent para generar use cases y “user stories” a partir de la especificación funcional en Markdown.
- **glossary_agent:** utilizado por el Functional Spec Writer Agent para elaborar el glosario del proyecto a partir del mismo documento, evitando sobrecargar su “system prompt” con tareas accesorias.

Este conjunto de herramientas desde funciones simples en Python hasta agentes especializados utilizados como “tools” evidencia una progresiva sofisticación del ecosistema, incrementando la eficiencia, modularidad y capacidad de colaboración de la arquitectura propuesta.

Tabla 3*Comparación de enfoques para la implementación de herramientas de agente*

Tipo de tool	Ventajas	Desventajas
Function Tool	Muy simple y fácil de crear y usar.	Limitada a herramientas determinísticas; no puede emplearse directamente en otros “frameworks” o agentes distintos de AutoGen.
MCP Server	Estándar ampliamente soportado por la mayoría de “frameworks” y agentes actuales, lo que garantiza portabilidad y reutilización.	Restringido a herramientas determinísticas; requiere la creación de un servidor e invocación explícita, lo que implica mayor complejidad que una función en Python.
Agent Tool	Permite construir herramientas inteligentes, generativas y adaptativas; no sobrecarga el “system prompt” del agente principal ni el contexto del <i>GroupChat</i> , dado que es visible solo para el agente que la invoca.	Requiere “prompt engineering” cuidadoso; no es portable y funciona únicamente en AutoGen; no recomendable para tareas determinísticas (e.g., cálculos matemáticos).

Nota. La tabla resume la evolución de los enfoques empleados en la construcción de herramientas de agente dentro del marco AutoGen, destacando ventajas y limitaciones en términos de simplicidad, portabilidad, flexibilidad y aplicabilidad.

Actualización agosto 2025:

Canal de audio (ASR/TTS) y “workaround” de AutoGen

Necesidad y diseño de la solución.

Para habilitar entrevistas por voz entre el interviewer_agent y la persona usuaria, fue preciso integrar reconocimiento automático del habla (ASR) y síntesis de voz (TTS). Dado que AutoGen no admite audio como entrada/salida a nivel de agente, se implementó un “workaround” de mediación: los agentes siguen procesando únicamente texto, mientras que una capa externa convierte la salida textual del entrevistador en audio (TTS) y transcribe el habla del

usuario a texto (ASR) antes de entregarla al agente. En términos operativos, el flujo se resume así:

1. **interviewer_agent** → (texto de la pregunta) → TTS → **audio** para el usuario.
2. **Usuario** → (respuesta en audio) → ASR → transcripción a texto → **interviewer_agent**.

Este desacoplamiento preserva la transparencia para los agentes (no perciben audio), a la vez que mantiene la trazabilidad textual en los registros de conversación.

Herramientas seleccionadas.

ASR — Whisper (OpenAI): Se empleó Whisper, un modelo *sequence-to-sequence* basado en Transformers entrenado con ~680 000 horas de supervisión multilingüe y multitarea, con capacidad de reconocimiento y traducción de voz, lo que aporta robustez en condiciones reales y escenarios multilingües (ruidos de fondo, acentos) (Radford et al., 2023) (OpenAI, 2024) (OpenAI, 2023).

TTS — Kokoro (82 M): Para la síntesis se utilizó Kokoro-82M, un modelo open-weight (~82 M parámetros, licencia Apache-2.0) optimizado para baja latencia y calidad comparable a sistemas de mayor tamaño, con soporte para múltiples voces/idiomas y despliegue flexible (local o servicio) (Hexgrad, 2024) (Kokoro TTS).

Justificación técnica: Ambos modelos son basados en Transformers, lo que facilitó una tubería homogénea (tokenización y flujos de inferencia similares) y un despliegue local con control de latencia/privacidad. Whisper aporta robustez multilingüe en ASR, mientras que Kokoro equilibra eficiencia y naturalidad en TTS.

Integración operativa (sin modificar AutoGen)

Encapsulamiento de E/S de voz. La capa ASR/TTS se ubica fuera de AutoGen. Los ”*prompts*” y respuestas de los agentes permanecen en texto; solo la capa externa realiza las conversiones audio↔texto.

Trazabilidad: La transcripción ASR es la que se registra en los artefactos de conversación (p. ej., `interview.jsonl`), garantizando auditoría y reproducibilidad del proceso.

Concurrencia y latencia: La orquestación mantiene el mismo ciclo de turnos (pregunta→respuesta) del *interviewer loop*, controlando la latencia principalmente en TTS/ASR.

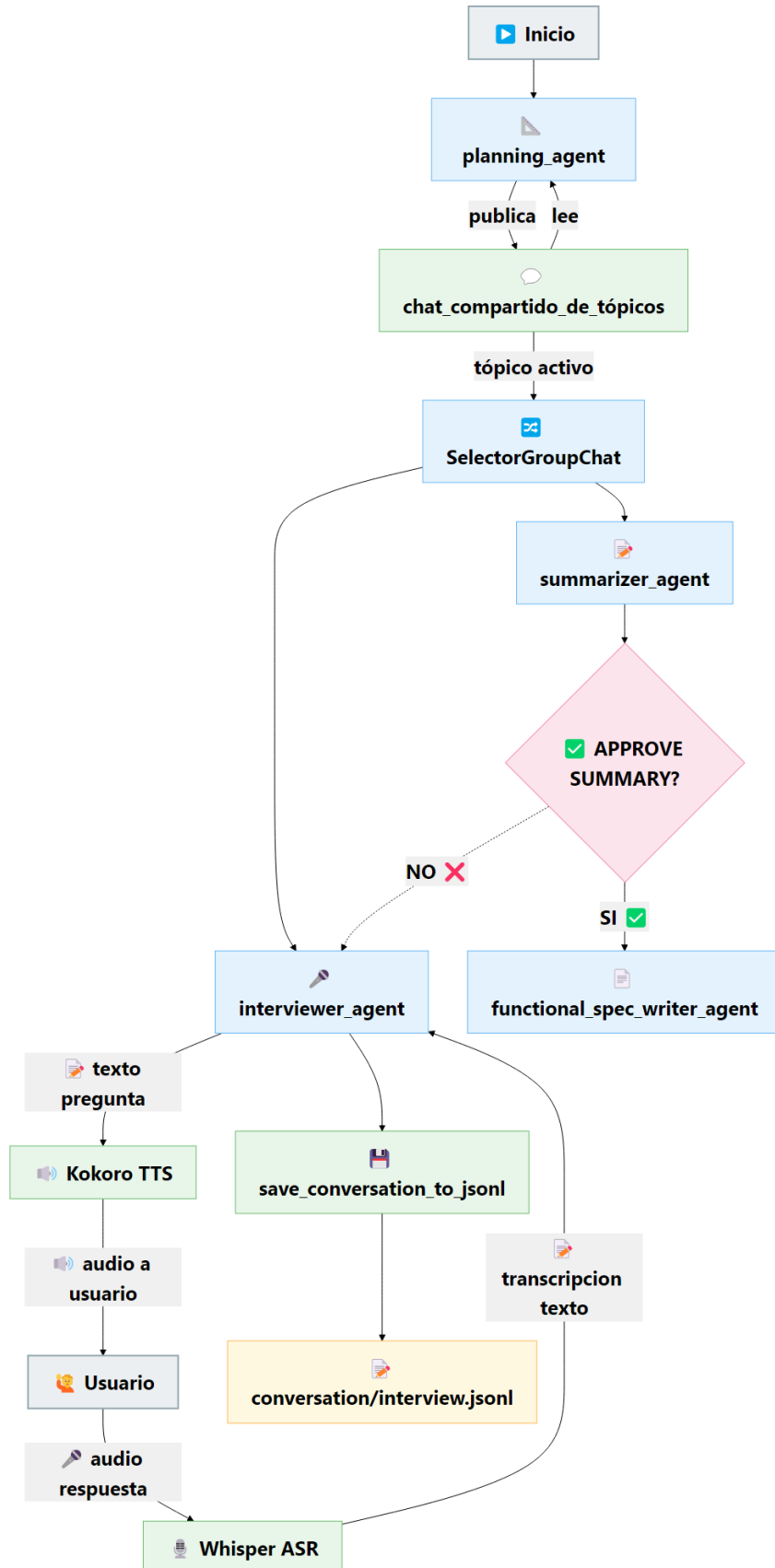
Portabilidad. Al no alterar la lógica de los agentes, el *workaround* es sustituible por otros ASR/TTS compatibles.

Consideraciones prácticas.

- **Calidad de audio:** La precisión del ASR mejora con audio limpio y microfonía consistente.
- **Idiomas y acentos:** Whisper ofrece reconocimiento multilingüe y language ID, útil en contextos bilingües o con cambio de código.
- **Privacidad:** Ambas piezas pueden operar localmente (según configuración), reduciendo exposición de datos sensibles.

Figura 21

Diagrama de Flujo “AudioBot” con AutoGen, Whisper y Kokoro TTS



“Audiobot” con modelos Realtime (speech-in/speech-out).

Novedad conceptual y motivación

A diferencia del flujo clásico ASR→LLM→TTS, los modelos “Realtime” de OpenAI (familia GPT-4o y su variante GPT-4o-mini “Realtime”) aceptan audio como entrada y generan audio como salida de forma nativa. Esto elimina la necesidad de encadenar un reconocedor (ASR) y un sintetizador (TTS) externos para cada turno, y habilita interacciones con menor latencia y más cercanas al tiempo real (“speech-in/speech-out”) (OpenAI, 2024a; OpenAI, s. f.-a).

Arquitectura adoptada

Para validar esta capacidad se empleó la “OpenAI Realtime API”, que proporciona sesiones persistentes con baja latencia, audio bidireccional y “streaming” mediante WebRTC (cliente) o WebSocket (servidor a servidor); además, soporta “function calling” y eventos en tiempo real (OpenAI, s. f.-b; OpenAI, s. f.-a). En nuestro caso, se seleccionó el modelo “GPT-4o-mini Realtime” por su eficiencia costo/rendimiento (OpenAI, s. f.-c; Microsoft, 2025).

Implicación para el “stack”. Como AutoGen no soporta audio en tiempo real a nivel de agentes, no existe un “workaround” transparente (a diferencia del caso ASR/TTS clásico, que envolvía texto). Por ello, la entrevista en tiempo real se implementó directamente con “OpenAI Realtime API + GPT-4o-mini Realtime” (interviewer y user proxy acoplados a la API de OpenAI), y el resultado (resumen/conclusiones) se inyecta después en el flujo AutoGen para continuar con `summarizer\_agent` → “APPROVE SUMMARY?” → writer → checker → diagram... igual que cuando cargamos un script de Teams (OpenAI, s. f.-b; Microsoft, 2025).

Flujo operativo

1. **Sesión “Realtime” (WebRTC/WebSocket) con GPT-4o-mini “Realtime”:** el usuario habla y recibe voz del modelo (speech-in/speech-out).
2. **Post-proceso:** se genera un resumen textual de la entrevista (por el propio modelo o por un paso auxiliar).
3. **Handoff a AutoGen:** el resumen se pasa al summarizer_agent y desde allí se sigue el diamante “APPROVE SUMMARY?” y el “pipeline” estándar (writer, checker con lazos de NEEDS_INFO, diagramación y export).

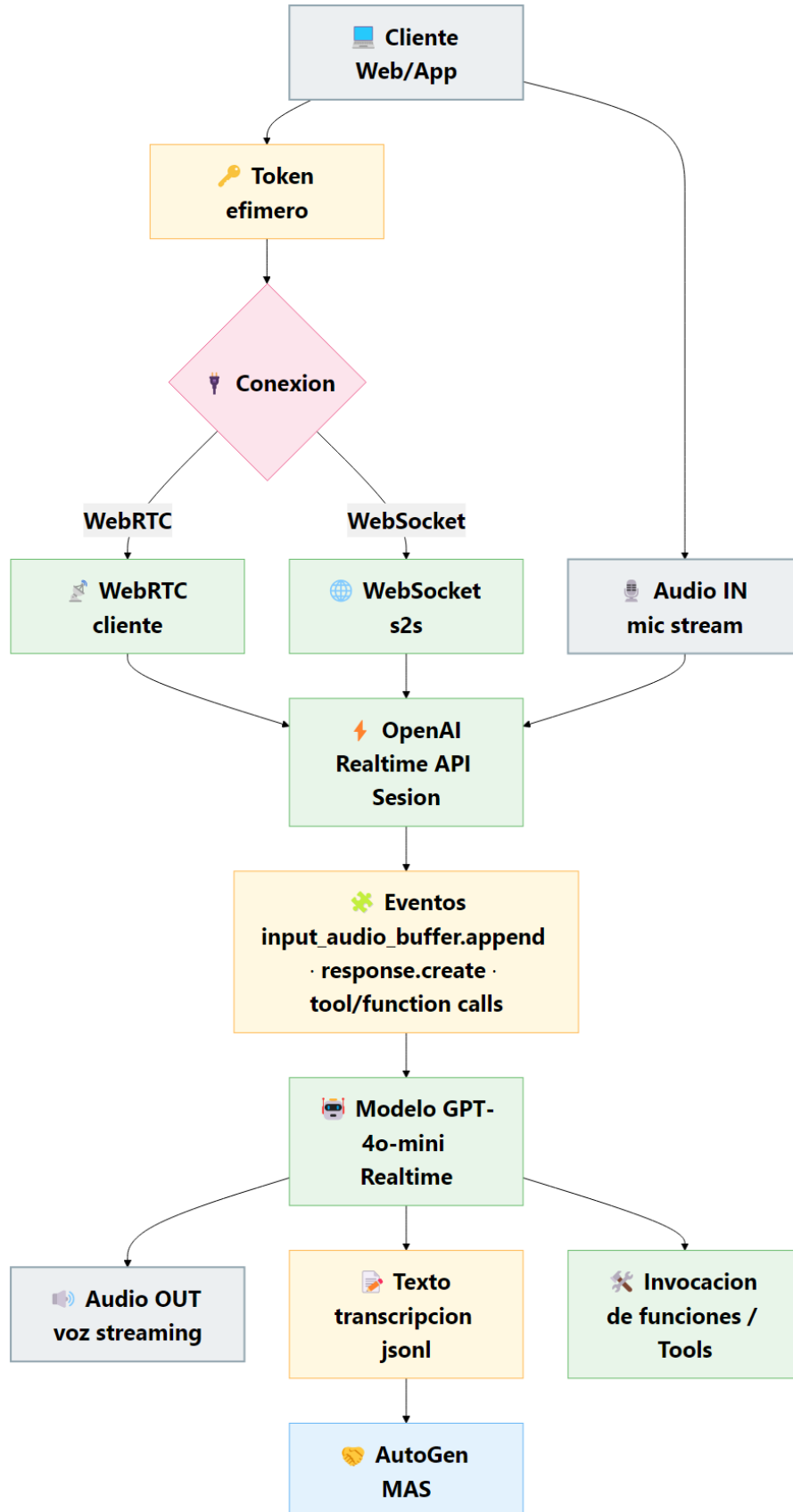
Este diseño reduce latencia y evita pérdidas de naturalidad conversacional al no requerir ASR/TTS intermedios, manteniendo al mismo tiempo la trazabilidad textual en el tramo AutoGen (OpenAI, 2024a; OpenAI, s. f.-a).

Consideraciones prácticas

- **Conexión:** WebRTC para cliente (audio playback inmediato) y WebSocket para integraciones “server-side”; uso de tokens efímeros en clientes (OpenAI, s. f.-b).
- **Modelo:** gpt-4o-mini-realtime-preview (versión 2024-12-17 en Azure/OpenAI), recomendado para pruebas de coste contenido (Microsoft, 2025).
- **Compatibilidad:** cuando se requiera persistir el histórico de la entrevista, se convierte a texto y se vuelca al flujo de AutoGen, análogo a cargar un guion de conversación.

Figura 22

Diagram de flujo “Audiobot” con “OpenAI Realtime API” y “GPT4-mini-realtime”



Entrevista telefónica con OpenAI Realtime + SIP (Twilio).

Objetivo y aporte

Con el fin de habilitar entrevistas por llamada telefónica (PSTN) sin perder las ventajas de "speech-in/speech-out" de los modelos "Realtime", se incorporó un ramal que enruta el audio de la llamada a la "OpenAI Realtime API" y devuelve audio de respuesta al/el interlocutor/a. La novedad es que el modelo ("GPT-4o-mini Realtime") procesa directamente audio como entrada y audio como salida, evitando la tubería ASR→LLM→TTS y reduciendo latencia percibida respecto de pipelines encadenados.

Arquitectura de integración

1. Entrada telefónica (PSTN/SIP) vía Twilio:

La llamada ingresa por "Twilio Programmable Voice" utilizando SIP Interface/SIP Domains (o PSTN directo). Twilio expone el audio crudo de la llamada a través de "Media Streams" y lo envía en tiempo real a un servidor WebSocket definido por la aplicación (mediante TwiML <Start><Stream> o <Connect><Stream>).

2. Puente a "OpenAI Realtime":

El servidor WebSocket actúa como bridge hacia la "OpenAI Realtime API", que soporta sesiones de baja latencia por WebRTC (cliente) o WebSocket (server-to-server). En nuestras pruebas operativas se utilizó "GPT-4o-mini Realtime" por su mejor relación costo-rendimiento.

3. Respuesta de audio hacia la llamada:

Con "Media Streams" bidireccionales, el servidor puede enviar audio de vuelta a Twilio para que sea inyectado en la llamada, cerrando el bucle de "speech-in/speech-out end-to-end". (Twilio documenta los tipos de mensajes y el flujo de ida y vuelta para "Streams").

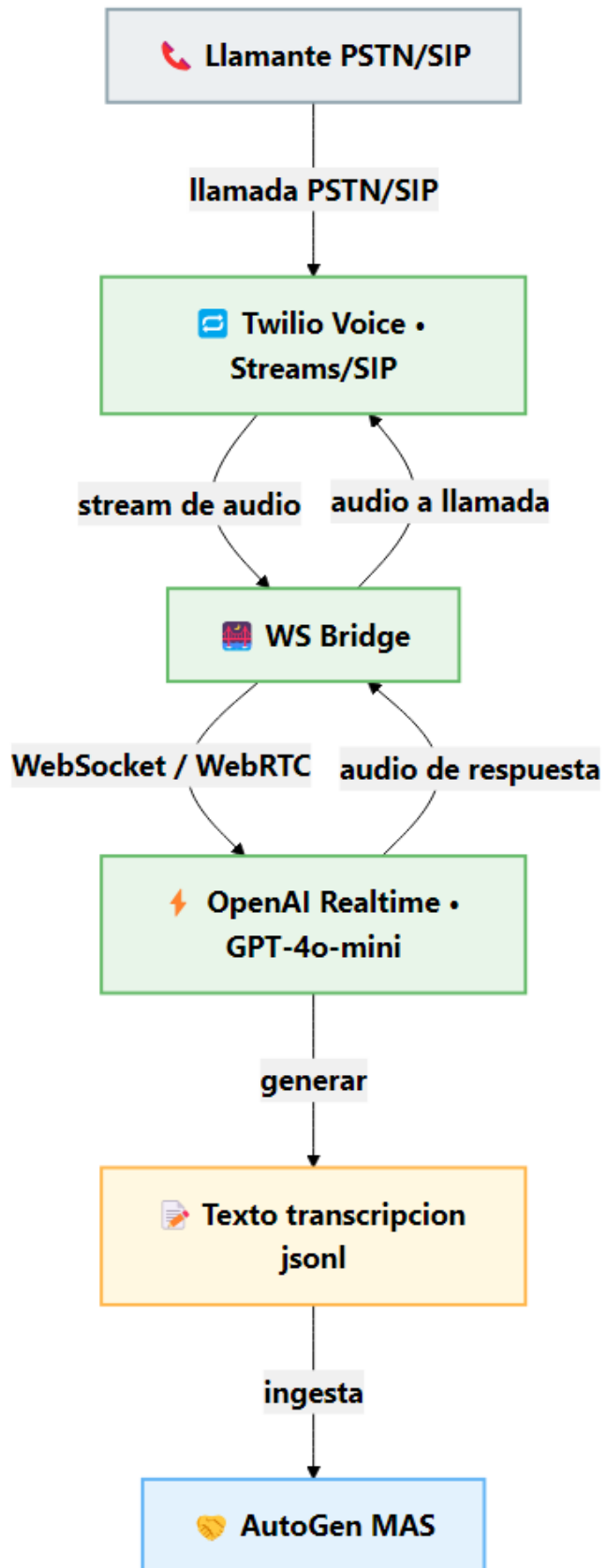
Operación dentro del flujo general

- La entrevista telefónica se realiza íntegramente en “Realtime” (modelo “GPT-4o-mini Realtime”), fuera de AutoGen (pues los agentes de AutoGen no soportan audio en tiempo real).
- Una vez concluida la llamada, se genera un resumen textual y se inyecta en el flujo de AutoGen de la misma manera que cuando cargamos un “conversation script” (Teams): el `planning_agent` indica al `SelectorGroupChat` enviar directamente al `summarizer_agent`, y desde allí se sigue el diamante “APPROVE SUMMARY?” y el pipeline estándar (`writer` → `checker` → `diagram` → `export`).

Notas de implementación

- **SIP/PBX:** Twilio SIP Interface permite registrar dominios SIP y enrutar llamadas desde/hacia infraestructuras SIP existentes (PBX/SBC), además de PSTN.
- **Streaming:** TwiML `<Start><Stream>/<Connect><Stream>` “forkea” el audio en near real-time hacia nuestro WebSocket; desde allí se reenvía al endpoint Realtime.
- **Bidireccionalidad:** los Media Streams admiten mensajes de ida y vuelta; Twilio documenta el formato de los frames y eventos (Connected, Start, Media, Stop, etc.).
- **Elección del transporte:** WebRTC suele ser preferible para clientes (latencia mínima), mientras que WebSocket es adecuado para server-to-server; ambos están soportados por “Realtime”.

Figura 23
Diagrama de flujo “AudioBot” y llamada



- **Entrada/salida de voz en tiempo real:** Twilio enruta el audio telefónico (PSTN/SIP) vía “Media Streams” a un “WS Bridge”, que conecta con “OpenAI Realtime” (“GPT-4o-mini Realtime”).
- **Sin ASR/TTS externos:** el modelo procesa audio→audio nativamente; al final se obtiene un resumen textual.
- **Handoff a AutoGen:** el resumen se entrega al `planning_agent`, que instruye a `SelectorGroupChat` enviar directo al `summarizer_agent` y continuar con “APPROVE SUMMARY?” (sí→writer; no→interviewer).

Finalmente, aunque no se considera un ítem específico de evaluación funcional, cabe destacar que todo el desarrollo del sistema se realizó siguiendo principios de diseño limpio (“Clean Code”). Se priorizó una arquitectura modular, con separación clara entre la lógica de negocio y las interfaces (tanto en “Jupyter Notebook” como en la interfaz web desarrollada con Chainlit).

El código fue escrito conforme a las convenciones de estilo PEP8, y verificado mediante herramientas como Pylint y Pylance para asegurar legibilidad, mantenibilidad y coherencia sintáctica. Asimismo, se implementó un esquema de configuración dinámica mediante archivos YAML, que facilita la gestión de múltiples aspectos del sistema:

- Definición y configuración de agentes, incluyendo sus “system messages” y herramientas disponibles.
- Estructura y participantes del SelectorGroupChat.
- Configuración del patrón Swarm, permitiendo modificar el flujo de delegación sin alterar la lógica interna.
- Declaración de modelos LLM disponibles (locales o “cloud-based”), permitiendo su activación selectiva en tiempo de ejecución.

Inclusión dinámica de nuevas fuentes web en el módulo RAG, con posibilidad de escalar el conocimiento del sistema sin reentrenar modelos.

Esta estrategia orientada a la modularidad y extensibilidad facilita futuras integraciones, como la incorporación de patrones de coordinación diferentes (por ejemplo, GraphFlow) o la experimentación con nuevos entornos de despliegue sin comprometer la integridad del código base.

Figura 24
UI del Business Analyst GTP

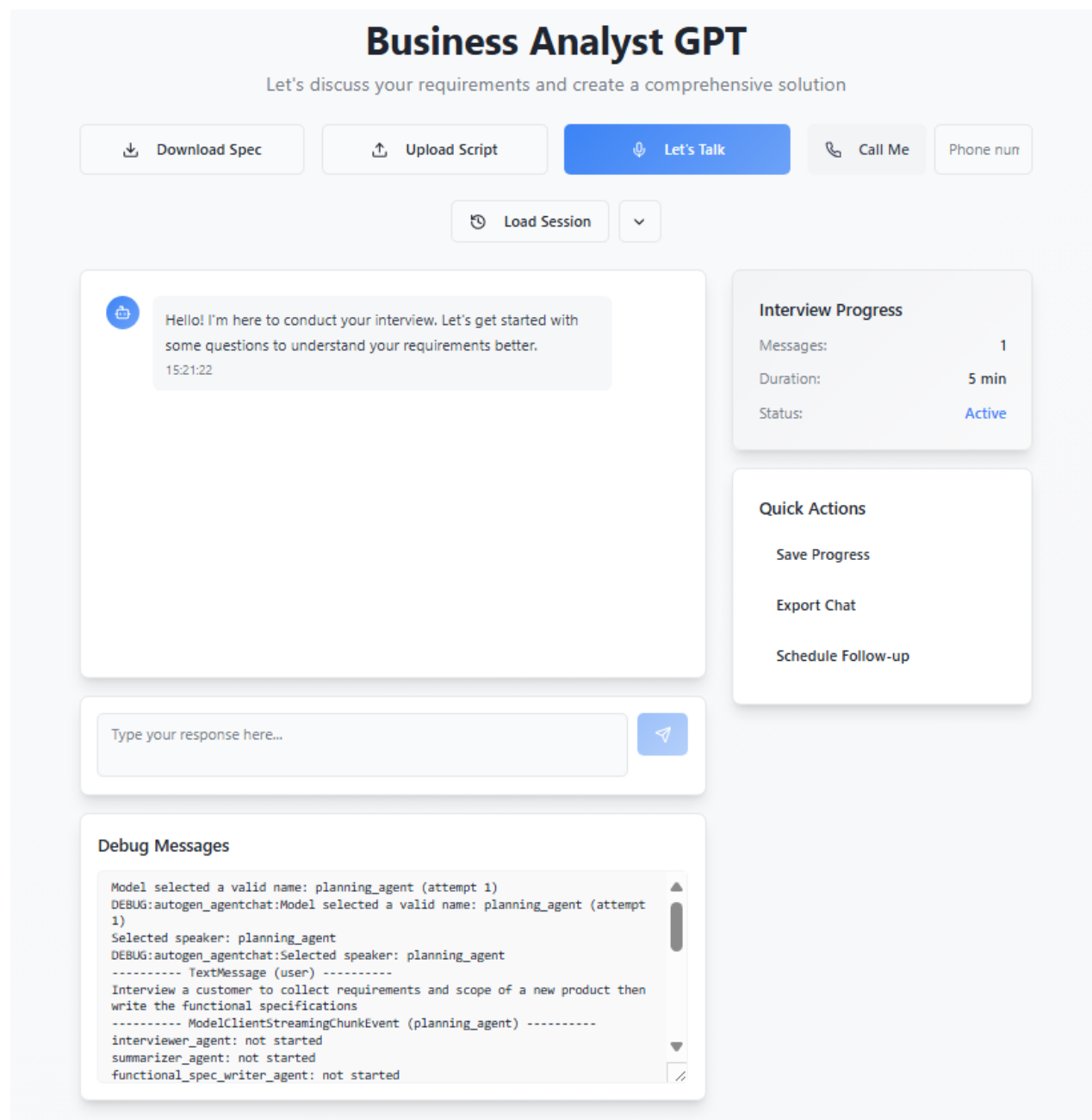
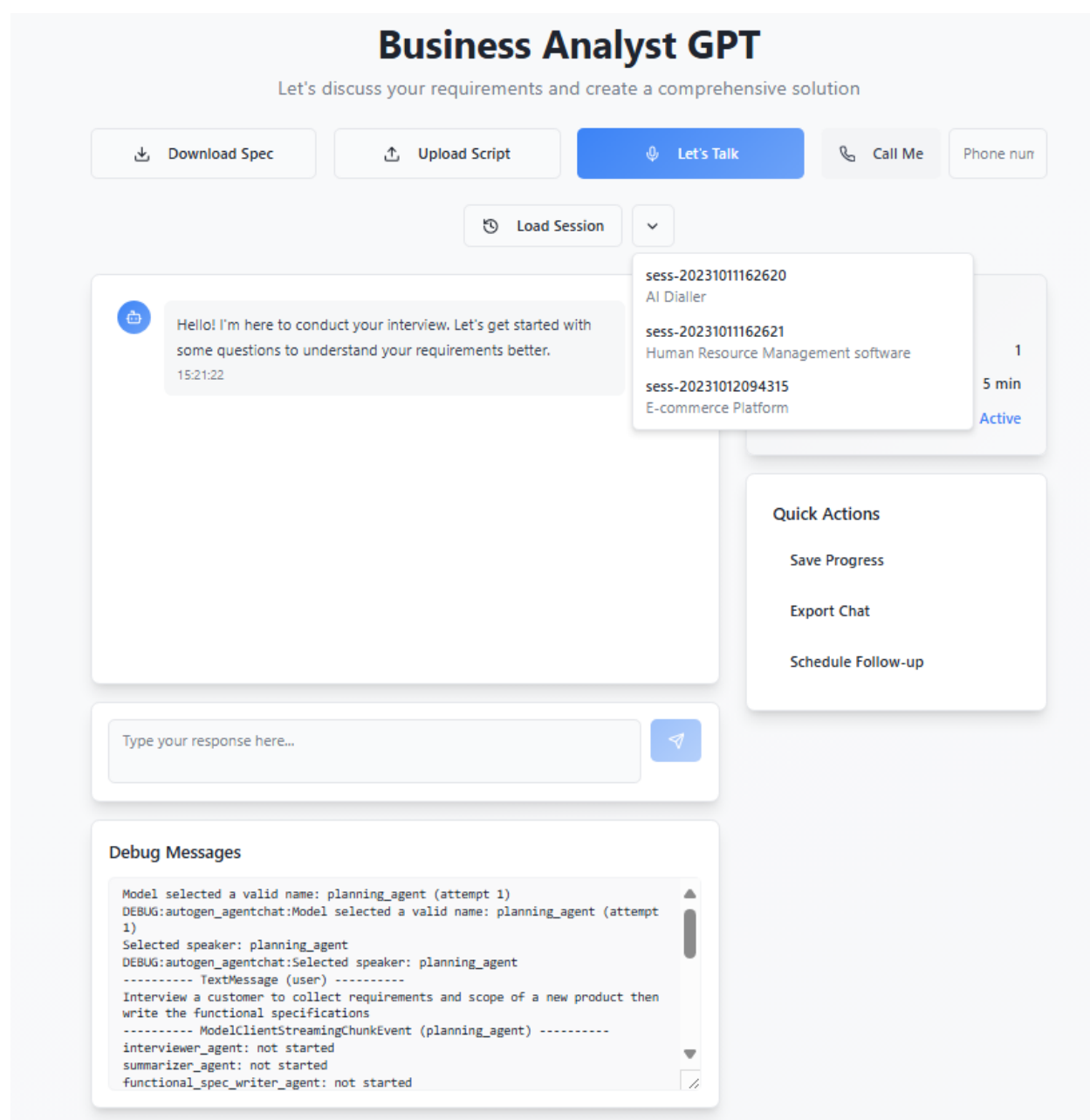


Figura 25

UI del Business Analyst GTP – Load Session dropdown Menu



Descripción funcional de la interfaz (vista “Business Analyst GPT”) — versión actualizada.

Acciones principales (barra superior)

- **Download Spec:** Descarga la especificación funcional consolidada en PDF.
- **Upload Script:** Carga un transcript externo (Teams/Zoom/Meet o resumen Realtime). Lo procesa `load_script_agent` y el `planning_agent` envía el paquete directamente al `summarizer_agent`.
- **Let’s Talk:** Inicia entrevista por voz (ASR/TTS o Realtime, según configuración).
- **Call Me:** Solicita llamada telefónica (SIP/PSTN vía Twilio) para realizar la entrevista por voz.
- **Load Session:** Restaura una sesión previa a partir de un `session_id`. Internamente invoca el flujo de carga de experiencia (p. ej., `load_experience_agent` + `load_experience_from_chromadb`) para traer los artefactos de `experience/<SESSION_ID>/` (entrevista, FSD, diagrama) al workspace y continuar donde se dejó.

Área de conversación (panel central)

- **Mensajes del entrevistador:** Muestra las preguntas del `interviewer_agent` con hora.
- **Campo de respuesta del usuario:** Entrada de texto para responder/aclarecer.
- **Botón de envío:** Registra la respuesta y avanza el flujo de orquestación.

Progreso de la entrevista (panel derecho)

- **Interview Progress:** Indicadores de mensajes, duración y estado (p. ej., Active).
- **Quick Actions.**
 - **Save Progress:** guarda el estado actual.
 - **Export Chat:** exporta el hilo para auditoría.
 - **Schedule Follow-up:** programa un seguimiento.

Consola técnica (“Debug Messages”)

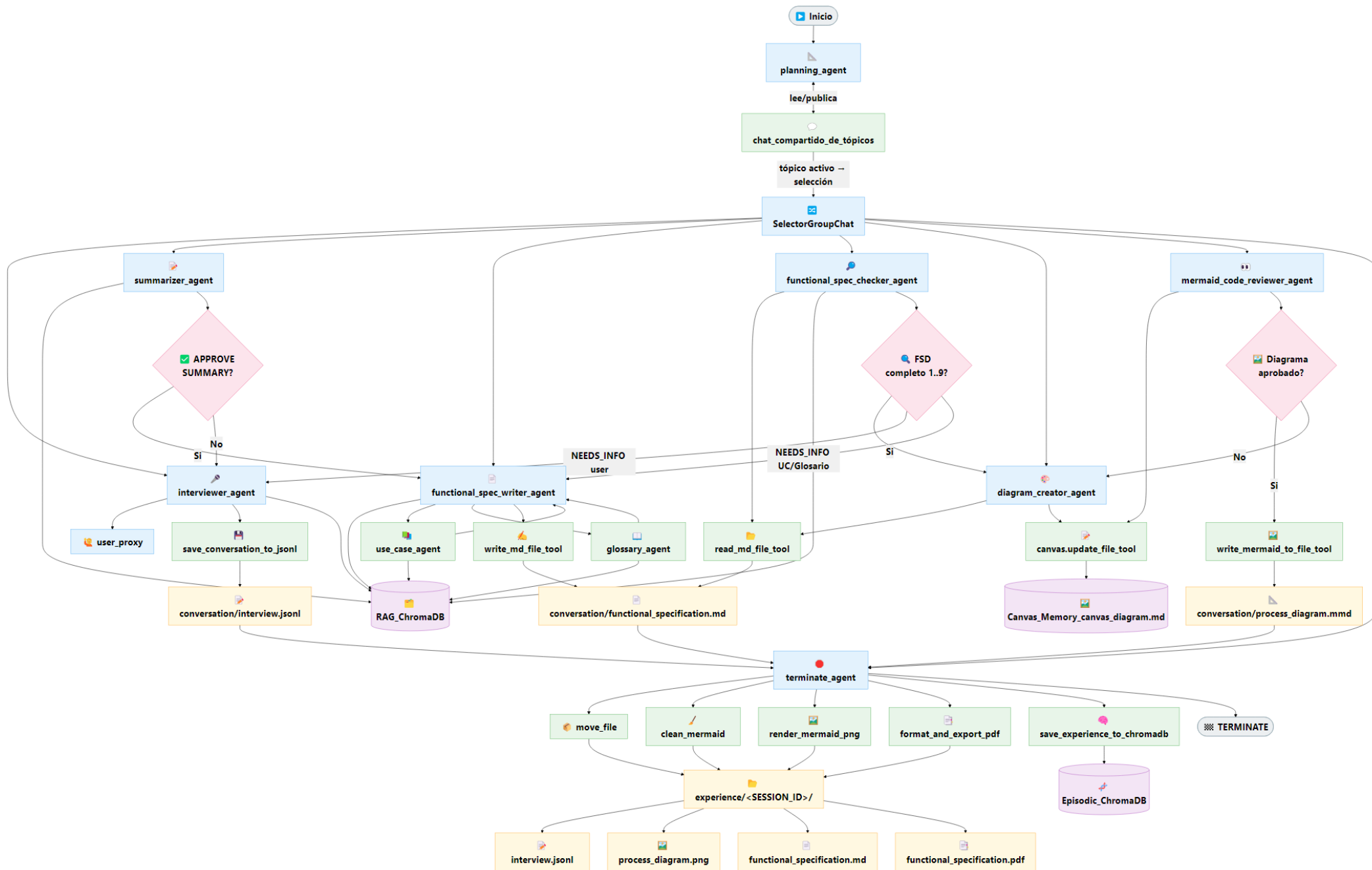
Traza la orquestación: agente seleccionado, tareas asignadas, marcadores (READY, NEEDS_INFO), etc., para diagnóstico y auditoría.

Flujos de uso típicos

- **Nueva entrevista por voz:** Let’s Talk / Call Me → entrevista → summarizer_agent → APPROVE SUMMARY? → writer → checker → diagrama → PDF.
- **Cargar transcript externo:** Upload Script → load_script_agent → planner/selector → summarizer_agent → APPROVE SUMMARY? → flujo normal.
- **Reanudar trabajo previo:** Load Session → (restaura artefactos vía ChromaDB/experiences) → planner/selector → continuar en el punto pendiente (p. ej., checker o diagram).

Figura 26

Diagrama de flujo de la arquitectura “end to end” para el caso de uso “Entrevista por chat”



Nota: debido a la complejidad del diagrama “end to end” se adjunta una versión animada pasa a paso con subtítulos explicativos. Hacer click en el link, dentro presionar siguiente para avanzar.

https://junmiron.github.io/Diagrama_end_to_end/

Figura 27

Diagrama de flujo de la arquitectura para el caso de uso “Cargar/Continuar una sesión anterior”

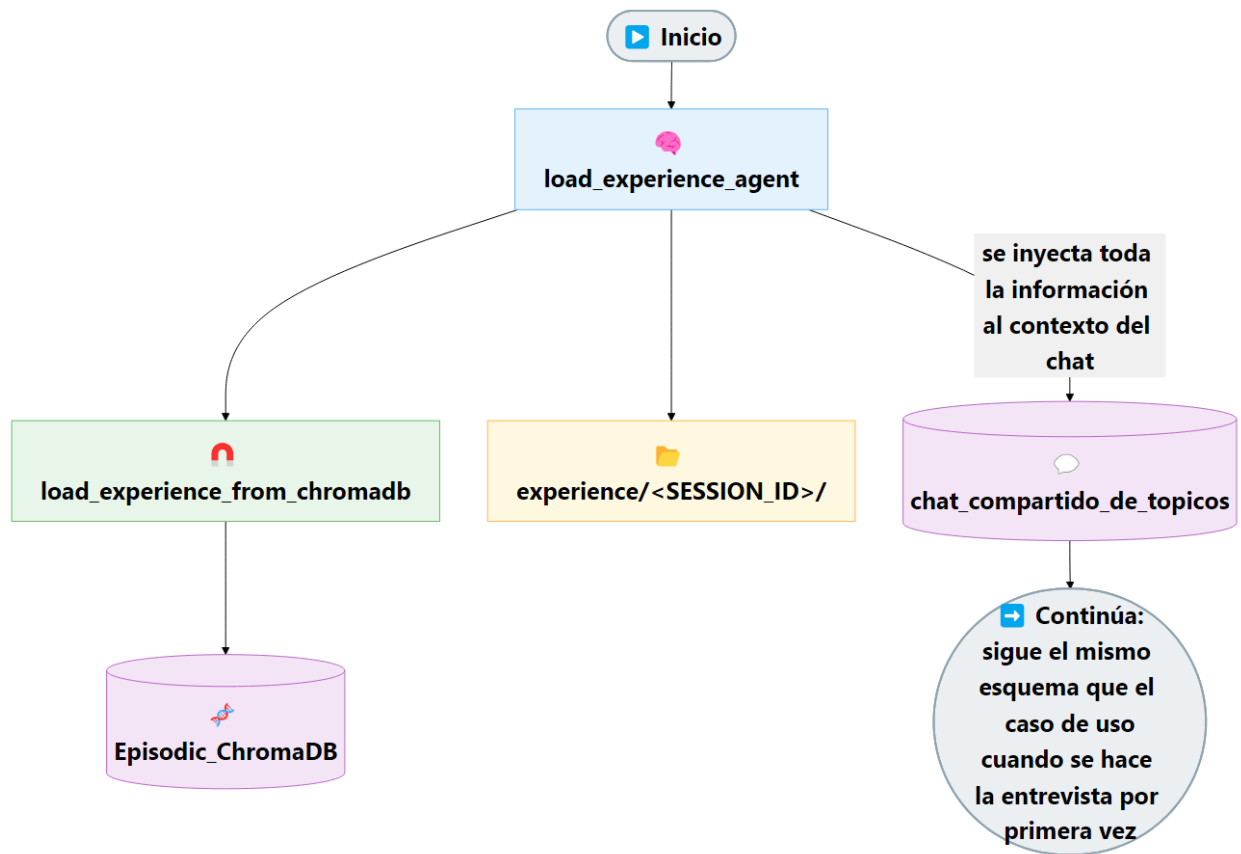
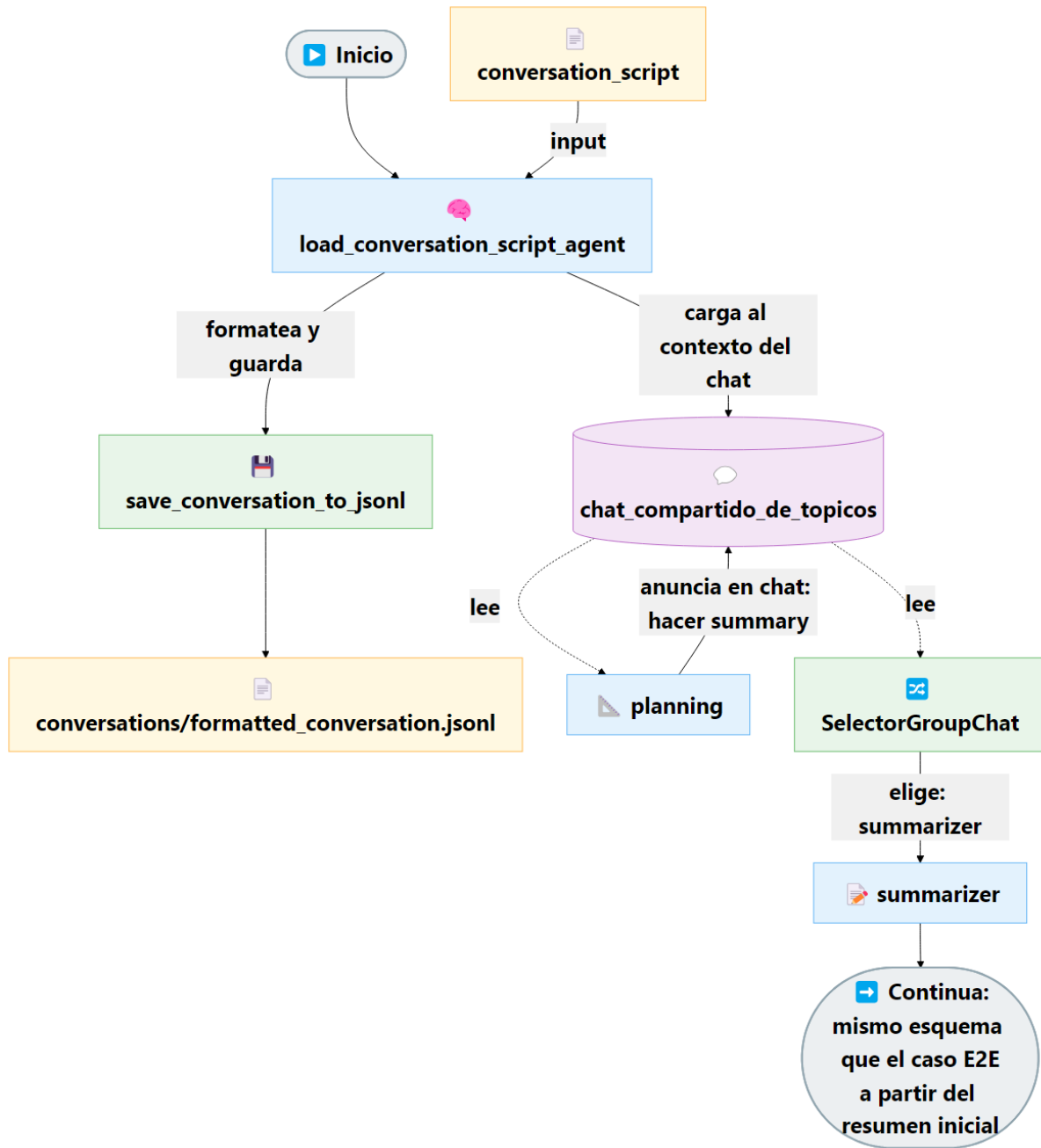


Figura 28

Diagrama de flujo de la arquitectura para el caso de uso “Carga de script de Teams/Slack/Whatsapp”



Diseño de pruebas

Se diseñaron tres casos reales que utilizaron agentes orquestados por AutoGen, *SelectorGroupChat* y *Swarm*:

- **AI Dialer:** un proyecto de chatbot orientado a ventas telefónicas.
- **Human Resource Management software:** sistema de gestión de recursos humanos.
- **Online Apparels Shopping Website:** plataforma de comercio electrónico para prendas.

En cada caso, los agentes entrevistaron usuarios, sintetizaron requisitos, generaron documentos funcionales, verificaron contenidos y crearon diagramas de proceso. Se validaron resultados ante usuarios, confirmándose:

- La viabilidad funcional del sistema en escenarios reales de negocio.
- La efectividad de la coordinación semántica y la trazabilidad en cada etapa.
- La flexibilidad en el uso de flujos de tratamiento y herramientas de debugging.

Evaluación y ajuste, con base en indicadores de eficiencia, satisfacción y coherencia semántica

La etapa de evaluación y ajuste del sistema se desarrolló considerando tres ejes clave: eficiencia operativa, satisfacción del usuario experto y coherencia semántica en la interacción y generación de contenido. Para ello, se implementaron mecanismos de observabilidad y trazabilidad, junto con instrumentos cualitativos de retroalimentación.

Logging y observabilidad

En cuanto a registro de actividad, AutoGen integra de forma nativa el módulo logging de Python, lo cual permitió capturar eventos relevantes de ejecución sin necesidad de instrumentación adicional. Esta funcionalidad facilitó el seguimiento del comportamiento de los agentes, especialmente en casos donde se requería depurar tareas interrumpidas o resultados inconsistentes.

Para el seguimiento estructurado de flujos, se utilizó el sistema de tracing y observabilidad incorporado en AutoGen, el cual está basado en la librería OpenTelemetry. Esta integración permitió:

- Recolectar trazas detalladas del recorrido de cada tarea entre agentes.
- Analizar el desempeño de herramientas y tiempos de respuesta en cada etapa.
- Comprender cómo se estructuraban las decisiones en flujos colaborativos complejos.

Las trazas fueron visualizadas a través de Jaeger UI/AI Toolkit Tracing, una interfaz gráfica que permitió explorar de manera intuitiva los caminos de ejecución, identificar cuellos de botella y rastrear errores a nivel de agente o herramienta.

AutoGen sigue las convenciones semánticas de OpenTelemetry tanto para agentes como para herramientas, e incorpora los lineamientos emergentes del grupo de trabajo de Semantic Conventions for GenAI Systems, lo que permite mantener un modelo de trazabilidad alineado con estándares modernos.

Indicadores evaluados

La evaluación del sistema se organizó en torno a tres indicadores principales:

- Eficiencia: se midió como el tiempo total necesario para completar un documento de especificación funcional (Functional Spec Document) desde la interacción inicial hasta la entrega final. Esta métrica permitió identificar variaciones de desempeño según el modelo utilizado, la complejidad del caso y la modalidad de orquestación empleada.
- Satisfacción del usuario experto: se aplicó una encuesta semiestructurada a un grupo de profesionales compuesto por dos “project managers” (PM), dos “business analysts” (BA), un “solution architect” (SA), un “platform manager” y un “operations manager”. Las preguntas abordaron aspectos como utilidad percibida, claridad de los resultados,

grado de automatización logrado y potencial de aplicación futura. La retroalimentación fue utilizada tanto para validar hipótesis del diseño como para priorizar mejoras funcionales.

Coherencia semántica: este criterio se evaluó a través de dos mecanismos:

Análisis manual de las trazas recolectadas (logs y spans), verificando si las transiciones entre agentes mantenían la intención del usuario original, sin pérdida de contexto o contradicciones internas.

Revisión cualitativa de las entrevistas entre el agente entrevistador y los usuarios humanos, dentro del esquema “*human-in-the-loop*”. Se analizó la pertinencia de las preguntas, la fidelidad del resumen generado y la alineación final del documento con lo expresado durante la interacción.

Este enfoque mixto cuantitativo y cualitativo, automático y manual permitió ajustar comportamientos de agentes, mejorar “prompts” internos y afinar mecanismos de validación, elevando la calidad global del sistema sin comprometer su trazabilidad ni su capacidad de auditoría.

4. Resultados

Este capítulo presenta y discute la evidencia empírica obtenida durante la evaluación piloto del sistema, con el objetivo de valorar su eficiencia operativa, su calidad semántica y la satisfacción de usuarios expertos. La evaluación se basó en tres casos de uso reales (AI Dialer, Human Resource Management software y Online Apparels Shopping Website) orquestados con AutoGen, SelectorGroupChat y Swarm, y se complementó con un caso real de actuación seleccionado (AI Dialer) para la comprobación de métricas y el seguimiento extremo a extremo de todo el flujo (entrevista, síntesis, especificación funcional, diagramación y exportación).

La evidencia procede de cuatro fuentes principales:

- 1. Tiempos de ciclo y trazabilidad** (registros de orquestación y archivos JSONL de entrevista),
- 2. Calidad semántica** (alineación pregunta–respuesta por materia y análisis de la traza de agentes),
- 3. Percepción experta** (21 encuestas semiestructuradas: 7 expertos $\times$ 3 productos), y
- 4. Calidad de artefactos** (documento de especificación funcional y diagramas de proceso generados).

Este enfoque de métodos mixtos (métricas cuantitativas + análisis cualitativo) permite triangular resultados y reforzar su validez interna.

En términos de procedimiento, el caso real de actuación se emplea como hilo conductor: primero para medir latencias y tiempos del ciclo pregunta→respuesta→persistencia, luego para evidenciar la coherencia temática y la cobertura del guion (dos preguntas por materia), y finalmente para ilustrar la calidad de los artefactos entregables (FSD y diagrama). De manera complementaria, las encuestas al panel experto (PM, BA, SA, Platform y Operations) aportan una valoración de utilidad, claridad, automatización y aplicabilidad en contextos reales.

La organización del capítulo es la siguiente: en 4.1 se reportan los ahorros de tiempo frente a ciclos tradicionales; en 4.2 se analiza la calidad semántica y capacidad responsiva, incorporando evidencia directa del JSONL y ejemplos anotados; se incluyen, además, análisis de la traza de agentes para describir la coordinación interna y el cumplimiento del protocolo; posteriormente se presentan los hallazgos sobre calidad funcional de diagramas y el impacto del formateo del PDF; y, finalmente, se sintetizan los beneficios globales y el seguimiento del caso real utilizado para la verificación integral del flujo.

En conjunto, los resultados muestran que, especialmente con modelos de alto rendimiento (p. ej., GPT-4o), el sistema reduce significativamente los tiempos de entrega, mantiene coherencia semántica en la entrevista y genera artefactos útiles y trazables, con niveles de satisfacción entre medios y altos en el panel experto. Los instrumentos, extractos y artefactos completos se documentan en los Apéndices para su consulta y auditoría.

4.1 Eficiencia Reducida de Semanas a Horas

- En ciclos anteriores, la creación de documentos de especificación funcional (Functional Spec) llevaba habitualmente entre cuatro y ocho semanas debido a múltiples reuniones y sesiones de revisión.
- Con BusinessAnalystGPT y flujos automatizados con AutoGen, este proceso se completó en cuestión de horas o como máximo pocos días, representando una reducción dramática de tiempo de entrega.
- Según el equipo de desarrollo, el ahorro en coordinación redujo un 70–80 % del tiempo requerido para consolidar requisitos.

4.2 Calidad Semántica y Capacidad Responsiva

- La calidad semántica del sistema fue excelente al utilizar GPT-4o: la terminología técnica, la estructura del documento y la cohesión de ideas fueron muy valoradas.
- La interfaz demostró una responder rápida e interactiva, sin retardos perceptibles, generando una experiencia fluida y natural.
- Las transcripciones y resúmenes automáticos fueron particularmente útiles para asegurar que la intención del usuario se mantuviera íntegra.

Calidad semántica de las conversaciones (evidencia con JSONL)

4.2.1 Propósito

Acreditar, con datos del archivo interview.jsonl, que las entrevistas guiadas por el interviewer_agent mantienen alineación temática, cumplimiento del protocolo y ritmo conversacional en tiempo casi real (baja latencia por ciclo pregunta→respuesta→persistencia).

4.2.2 Evidencias extraídas del JSONL

A partir del archivo adjunto (session_id = sess-20231011162620), se obtuvieron los siguientes indicadores cuantitativos:

1. Cobertura del guion
 - 8 sujetos recorridos en el orden previsto: Product Overview; KPI & Success Metrics; Scope In and Out; Non-Functional Requirements; User Roles & Personas; Integrations & External Systems; Constraints & Assumptions; Dependencies & Risks.
 - 2 preguntas por sujeto (máximo definido por protocolo): 100 % de cumplimiento.
2. Conformidad del protocolo I/O

- 16 mensajes del entrevistador, todos comienzan con QUESTION:.
- Registro pareado y coherente: cada QUESTION: está seguido por la respuesta del usuario con el mismo subject y el mismo q_index $\rightarrow$ 100 % de alineación pregunta-respuesta.

3. Ritmo conversacional (latencia por ciclo)

- Duración total de la entrevista (primera a última marca temporal): ~15,96 min.
- Intervalo entre pares consecutivos pregunta-respuesta (medido sobre marcas de guardado en JSONL, es decir, ciclo completo):
 - a. Mediana: 54,0 s
 - b. Media: 63,8 s
 - c. Rango: 20,8 s (mín) – 135,9 s (máx)
- Interpretación: la conversación fluye en tiempos de interacción naturales (decenas de segundos por ciclo), compatibles con “near-realtime” en entrevistas guiadas (lectura de la pregunta, respuesta del usuario, y persistencia atómica del par en el registro).

Nota metodológica: el JSONL almacena cada par pregunta-respuesta de forma atómica; por ello, la latencia que aquí se reporta corresponde al ciclo completo (preguntar $\rightarrow$ responder $\rightarrow$ guardar), que es el indicador operativo relevante para evaluar fluidez en una entrevista.

4.2.3 Ejemplo ilustrativo (extracto JSONL)

A continuación, dos pares reales (texto tal cual en el archivo), que muestran pertinencia temática y utilidad de las preguntas:

- Product Overview — q_index=1
 - Interviewer: QUESTION: Can you briefly describe the new product and its primary purpose?

- User: The new product is an AI dialer that calls customers to offer ... the AI attempts to convince the customer up to three additional times before ending the call
- Non-Functional Requirements — q_index=1
 - Interviewer: QUESTION: Are there any specific regulatory requirements or compliance standards the system must adhere to?
 - User: Performance: the system should be able to support up to 300 ca... response must be in real time, and use natural language and human like

Estos ejemplos reflejan claridad y especificidad en las formulaciones del entrevistador (quién/qué/para qué), y respuestas que el sistema puede trasladar con fidelidad a la especificación funcional (requisitos verificables y no funcionales).

4.2.4 Conclusión

El registro JSONL confirma que las conversaciones son semánticamente consistentes (alineación pregunta–respuesta por sujeto y por índice), cumplen el protocolo estricto de entrevista y presentan un ritmo de interacción cercano al tiempo real, adecuado para la recopilación eficaz de requisitos y su posterior consolidación en el documento de especificación funcional.

Figura 29
Extracto Jsonl proyecto AI Dialer

```
ce > sess-20231011162620 > {} interview.jsonl
{"_meta": {"session_id": "sess-20231011162620", "ts": "2025-08-09T20:35:33.157973Z", "n_records": 2}}
{"speaker": "interviewer", "message": "QUESTION: Can you briefly describe the new product and its primary purpose?", "subject": "Product Overview", "q_index": 1, "timestamp": "2025-08-09T20:35:33.157973Z"}
{"speaker": "user", "message": "The new product is an AI dialer that calls customers to offer travel packages. If the customer accepts, the call transfers to a human. If declined, the call is ended.", "subject": "Product Overview", "q_index": 1, "timestamp": "2025-08-09T20:36:12.337556Z"}
{"_meta": {"session_id": "sess-20231011162620", "ts": "2025-08-09T20:36:12.337556Z", "n_records": 2}}
{"speaker": "interviewer", "message": "QUESTION: What are the key features or functionalities that you envision for this AI dialer?", "subject": "Product Overview", "q_index": 2, "timestamp": "2025-08-09T20:36:12.337556Z"}
{"speaker": "user", "message": "scope in: the dialer will be integrated with the call center system to make and transfer calls, with the CRM to get customer information like name, email, phone number, etc.", "subject": "Product Overview", "q_index": 2, "timestamp": "2025-08-09T20:36:40.339321Z"}
{"_meta": {"session_id": "sess-20231011162620", "ts": "2025-08-09T20:36:40.339321Z", "n_records": 2}}
{"speaker": "interviewer", "message": "QUESTION: How would you define success for this project in terms of measurable business objectives?", "subject": "KPI & Success Metrics", "q_index": 3, "timestamp": "2025-08-09T20:37:13.210804Z"}
{"speaker": "user", "message": "the KPI will be configurable and metrics will include successful calls, error calls, successful deal, rejected deals, the report should include the number of calls made, the number of calls answered, the number of calls transferred to a human, the number of calls that resulted in a sale, etc.", "subject": "KPI & Success Metrics", "q_index": 3, "timestamp": "2025-08-09T20:37:13.210804Z"}
{"_meta": {"session_id": "sess-20231011162620", "ts": "2025-08-09T20:37:13.210804Z", "n_records": 2}}
{"speaker": "interviewer", "message": "QUESTION: Should additional reporting capabilities or real-time data visualization options be part of the MVP?", "subject": "KPI & Success Metrics", "q_index": 4, "timestamp": "2025-08-09T20:37:13.210804Z"}
{"speaker": "user", "message": "no", "subject": "KPI & Success Metrics", "q_index": 4, "timestamp": "2025-08-09T20:37:13.210804Z"}
{"_meta": {"session_id": "sess-20231011162620", "ts": "2025-08-09T20:37:44.574598Z", "n_records": 2}}
{"speaker": "interviewer", "message": "QUESTION: Are there any functionalities or integrations that can be delayed for future versions?", "subject": "Scope In and Out", "q_index": 5, "timestamp": "2025-08-09T20:38:20.921468Z"}
{"speaker": "user", "message": "scope in: the dialer will be integrated with the call center system to make and transfer calls, with the CRM to get customer information like name, email, phone number, etc.", "subject": "Scope In and Out", "q_index": 5, "timestamp": "2025-08-09T20:38:20.921468Z"}
{"_meta": {"session_id": "sess-20231011162620", "ts": "2025-08-09T20:38:20.921468Z", "n_records": 2}}
{"speaker": "interviewer", "message": "QUESTION: Are there specific use cases or customer demographics the product should prioritize first?", "subject": "Scope In and Out", "q_index": 6, "timestamp": "2025-08-09T20:38:20.921468Z"}
{"speaker": "user", "message": "That will be configurable and will depend on each campaign", "subject": "Scope In and Out", "q_index": 6, "timestamp": "2025-08-09T20:38:20.921468Z"}
{"_meta": {"session_id": "sess-20231011162620", "ts": "2025-08-09T20:38:55.328313Z", "n_records": 2}}
{"speaker": "interviewer", "message": "QUESTION: Are there any specific regulatory requirements or compliance standards the system must adhere to?", "subject": "Non-Functional Requirements", "q_index": 7, "timestamp": "2025-08-09T20:41:11.244248Z"}
{"speaker": "user", "message": "Performance: the system should be able to support up to 300 call simultaneously, Reliability: have an availability of 99.96%, responses have to be fast, Security: the system should be secure, etc.", "subject": "Non-Functional Requirements", "q_index": 7, "timestamp": "2025-08-09T20:41:11.244248Z"}
{"_meta": {"session_id": "sess-20231011162620", "ts": "2025-08-09T20:41:32.082905Z", "n_records": 2}}
{"speaker": "interviewer", "message": "QUESTION: Can you describe the main user roles and their interactions with the AI dialer?", "subject": "User Roles & Personas", "q_index": 8, "timestamp": "2025-08-09T20:41:32.082905Z"}
{"_meta": {"session_id": "sess-20231011162620", "ts": "2025-08-09T20:41:32.082905Z", "n_records": 2}}
```

Nota. Se utiliza “screenshot” ya que los textos son muy largos a lo ancho y a pegarlos completos se hace muy difícil la lectura. Para ver parte del texto ver Apéndice E

4.2.5 Caso real “AI Dialer” - Presentación de la evidencia conversacional

A continuación, se incorpora la transcripción resumida de la entrevista utilizada como fuente primaria para el caso “AI Dialer”. El material procede del archivo interview.jsonl y ha sido normalizado para lectura humana: se presentan pares Pregunta–Respuesta en orden cronológico de ocurrencia, con el “Subject” temático y, cuando corresponde, el índice de pregunta (q_index). Con el fin de preservar la claridad y la confidencialidad, se han omitido marcas temporales, identificadores técnicos, trazas internas y metadatos operativos; asimismo, se han aplicado ajustes ortotipográficos menores (sin alterar el contenido semántico) y anonimización cuando fue necesario. La terminología de negocio se mantiene íntegra para facilitar la trazabilidad con la Especificación Funcional y con los resultados del Capítulo.

Criterios de presentación:

- **Formato:** Subject → Pregunta → Respuesta, en secuencia cronológica.
- **Alcance:** se conserva la intencionalidad de cada intervención; no se incluyen elementos irrelevantes para la comprensión del caso.
- **Coherencia metodológica:** el contenido refleja el protocolo de entrevista adoptado (dos preguntas por materia), cuya evaluación cuantitativa y de latencia ya se reporta en secciones previas del capítulo.
- **Trazabilidad ampliada:** el registro completo JSONL se documentan en Apéndices E, a efectos de auditoría y reproducibilidad.

Subject: Product Overview (q_index=1)

Pregunta: Can you briefly describe the new product and its primary purpose?

Respuesta: The new product is an AI dialer that calls customers to offer travel packages. If the customer accepts, the call transfers to a human. If declined, the AI attempts to convince the customer up to three additional times before ending the call

Subject: Product Overview (q_index=2)

Pregunta: What are the key features or functionalities that you envision for this AI dialer?

Respuesta: scope in: the dialer will be integrated with the call center system to make and transfer calls, with the CRM to get customer information like name, phone number, location, salary, with a database to get the pre defined offers and to save calls results as successful call, closed deals, rejected deals, call errors and a RAG to answer customer doubts and

questions, t the end of the shift the AI should make a report with metrics and KPIs, scope out: whats app integration, social network integration

Subject: KPI & Success Metrics (q_index=1)

Pregunta: How would you define success for this project in terms of measurable business objectives?

Respuesta: the KPI will be configurable and metrics will include successful calls, error calls, successful deal, rejected deals, the report should include chats, tables and a dynamic dashboard will all KPI and metrics,

Performance: the system should be able to support up to 300 call

simultaneously, Reliability: have an availability of 99.96%, responses have to be in real time, and use natural language and human like voice quality,

Usability: use best practices, Security: use access control, dev full access to app and code, marketing to reports

Subject: KPI & Success Metrics (q_index=2)

Pregunta: Should additional reporting capabilities or real-time data visualization options be part of the MVP?

Respuesta: no

Subject: Scope In and Out (q_index=1)

Pregunta: Are there any functionalities or integrations that can be delayed for future versions?

Respuesta: scope in: the dialer will be integrated with the call center system to make and transfer calls, with the CRM to get customer information like name, phone number, location, salary, with a database to get the pre-defined offers and to save calls results as successful call, closed deals, rejected deals, call errors and a RAG to answer customer doubts and

questions, t the end of the shift the AI should make a report with metrics and KPIs

Subject: Scope In and Out (q_index=2)

Pregunta: Are there specific use cases or customer demographics the product should prioritize first?

Respuesta: That will be configurable and will depend on each campaign

Subject: Non-Functional Requirements (q_index=1)

Pregunta: Are there any specific regulatory requirements or compliance standards the system must adhere to?

Respuesta: Performance: the system should be able to support up to 300 call simultaneously, Reliability: have an availability of 99.96%, responses have to be in real time, and use natural language and human like voice quality, Usability: use best practices, Security: use access control, dev full access to app and code, marketing to reports

Subject: Non-Functional Requirements (q_index=2)

Pregunta: Are there additional constraints like language support, accessibility, or anything else we should consider?

Respuesta: it will be only English, voice quality has to be human like level, the tone has to be respectful but convincing

Subject: User Roles & Personas (q_index=1)

Pregunta: Can you describe the main user roles and their interactions with the AI dialer?

Respuesta: the users will be Travel agencies and airlines, the project stake holders are, marketing team provide pre-defined offers and pre-defined answers and documentation and past surveys for RAG, marketing VP is the

sponsor, It team for support and development, human agents to take transfer and close deals are testers, and the final customer from the companies that will receive the call

Subject: User Roles & Personas (q_index=2)

Pregunta: Are there specific user personas or scenarios that need to be addressed in the system design?

Respuesta: what I already told

Subject: Integrations & External Systems (q_index=1)

Pregunta: What systems or external integrations are necessary for the AI dialer to function effectively?

Respuesta: the dialer will be integrated with the call center system to make and transfer calls, with the CRM to get customer information like name, phone number, location, salary, with a database to get the pre defined offers and to save calls results

Subject: Integrations & External Systems (q_index=2)

Pregunta: Are there any additional external tools or APIs that may be optional but beneficial for the system?

Respuesta: WhatsApp support or any other social networks, but this is out of scope for this release

Subject: Constraints & Assumptions (q_index=1)

Pregunta: Are there any legal, regulatory compliance, or environmental constraints shaping this system's assumptions or functionality?

Respuesta: yes it has to be complaint with GDPR, PIII and all data has to be encrypted also calls will be done from 8am to 5pm only, all offers will be predefined and available in a database, this offer can be updated by the

marketing team as desired, There will be predefined responses in the database and also additional information of previous experiences in the RAG that the AI dialler could use

Subject: Constraints & Assumptions (q_index=2)

Pregunta: Are there any assumptions about resource availability, stakeholder responsibilities, or system limitations we should document?

Respuesta: no

Subject: Dependencies & Risks (q_index=1)

Pregunta: What are the potential risks or dependencies we should be aware of for this project?

Respuesta: we depend on cloud providers to host the dialer and SIP trunk providers for the calls

Subject: Dependencies & Risks (q_index=2)

Pregunta: How will these dependencies on cloud and SIP trunk providers be managed to avoid downtime or disruptions?

Respuesta: yes we have two different providers for cloud and two for SIP trunks

Cobertura: 8 subjects con 2 preguntas cada uno → 16 pares pregunta-respuesta

(coincide con el protocolo 2×8).

Longitud media: preguntas $\approx 14,5$ palabras; respuestas $\approx 37,7$ palabras → respuestas concretas y accionables para volcar a FSD.

Distribución por tema: balance 2/subject en: Product Overview; KPI & Success Metrics; Scope In and Out; Non-Functional Requirements; User Roles & Personas; Integrations & External Systems; Constraints & Assumptions; Dependencies & Risks.

4.2.6 Análisis de la traza de agentes: calidad semántica y seguimiento de preguntas

Objetivo. Evaluar, a partir de la traza ejecutiva de AutoGen/SelectorGroupChat, (a) la calidad semántica de las interacciones internas entre agentes y (b) el cumplimiento del protocolo de entrevista (orden de materias y máximo de dos preguntas por materia).

a) Cumplimiento del protocolo y trazabilidad

- Formato I/O estricto del entrevistador. Las intervenciones siguen el patrón requerido (una línea iniciada en QUESTION: por turno). Ejemplos representativos: KPI (“Should additional reporting...”), Alcance (“What functionalities or integrations are absolutely essential...”), Restricciones (“Are there any assumptions...”).
- Persistencia atómica de Q&A. Tras cada respuesta del usuario, el sistema invoca una única escritura con “Saved 2 record(s) to ./conversation/interview.jsonl”, dejando evidencia repetida y consistente de guardado por par pregunta–respuesta.
- Gestión de turnos y gating al usuario. El planner asigna explícitamente la respuesta al usuario y activa awaiting_user_reply=true, garantizando que ningún otro agente intervenga hasta recibir la contestación.

b) Cobertura por materias y seguimiento de preguntas

- Evolución del estado. La traza muestra incrementos graduales (instantáneas intermedias) desde materias iniciales hasta completar las restantes, respetando el tope de 2 preguntas por materia. Ver, por ejemplo, estados parciales (inicio con

solo Product Overview completado; luego KPI/Scope; posteriormente Integraciones) y estado final con ocho materias en done (asked=2).

- Ejemplos de alineación semántica pregunta–materia.
 - KPI & Success Metrics: confirmación sobre visualización en tiempo real (alineada a métricas).
 - Scope In/Out: priorización de funcionalidades e integraciones esenciales (alineada al alcance).
 - Integrations & External Systems: consulta sobre APIs/herramientas opcionales (alineada a integraciones).
 - Dependencies & Risks: registro de Q&A específico (evidencia de tratamiento del tema).
- Conclusión de la entrevista. El entrevistador emite un STATUS final con las ocho materias en done y asked=2, lo que confirma cierre completo y homogéneo.

c) Ritmo conversacional (latencia percibida)

- Las marcas de tiempo de los eventos indican latencias sub-minuto durante fases activas (p. ej., dos escrituras consecutivas a las 20:37:13 y 20:37:44 UTC), y progresión fluida hasta el cierre a las 20:51:30 UTC. Esto respalda una experiencia de entrevista interactiva en near real-time.

d) Observaciones operativas

- Selección de oradores estable. SelectorGroupChat elige oradores válidos y el planner coordina asignaciones sin desvíos del flujo definido, pese a warnings de mapeo de modelo que no afectan la lógica de turnos/estado.

Síntesis. La traza evidencia alta calidad semántica (preguntas pertinentes por materia) y fidelidad al protocolo (2 preguntas por tema, guardado atómico, gating al usuario). El cierre con las ocho materias completas sostiene una cobertura 100 % del guion de entrevista y una base sólida para la síntesis y elaboración posterior de la especificación funcional.

Figura 30
Extracto de la traza de agentes

```

ion > agent trace.txt

QUESTION: What are the primary metrics or KPIs you would use to measure the success of the AI dialer?
/run/media/jun/Jun 6T/Code/TFM_BA_GPT/venv/lib/python3.13/site-packages/autogen_agentchat/teams/_group_chat/_selector_group_chat.py:269: UserWarning: Resolved model mismatch
  response = await self._model_client.create(messages=select_speaker_messages)
Model selected a valid name: interviewer_agent (attempt 1)
DEBUG:autogen_agentchat:Model selected a valid name: interviewer_agent (attempt 1)
Selected speaker: interviewer_agent
DEBUG:autogen_agentchat:Selected speaker: interviewer_agent
----- MemoryQueryEvent (interviewer_agent) -----
[MemoryContent(content='• \nRoutine diagnostic checks \n5. Reliability \nThese requirements address product efficiency in the sense that \nthere are no wasted or duplicated
----- ModelClientStreamingChunkEvent (interviewer_agent) -----
QUESTION: How would you define success for this project in terms of measurable business objectives?
----- TextMessage (user) -----
the KPI will be configurable and metrics will include succesfull calls, error calls, succesfull deal, rejected deals, the reprot should include chats, tables and a dynamic
----- MemoryQueryEvent (interviewer_agent) -----
[MemoryContent(content='User and Customer Needs: \n \nWho are the end users/customers affected by this project? \n \nWhat are their specific needs, preferences, and pain po
id='8982a0b3-400b-4617-9f84-25720903a4c8' source='interviewer_agent' models_usage=RequestUsage(prompt_tokens=0, completion_tokens=0) metadata={} created_at=datetime.datetime(2025, 8, 9, 20, 36, 40, tzinfo=datetime.timezone.utc)
id='8982a0b3-400b-4617-9f84-25720903a4c8' source='interviewer_agent' models_usage=RequestUsage(prompt_tokens=0, completion_tokens=0) metadata={} created_at=datetime.datetime(2025, 8, 9, 20, 36, 40, tzinfo=datetime.timezone.utc)
DEBUG:autogen_agentchat.events:id='8982a0b3-400b-4617-9f84-25720903a4c8' source='interviewer_agent' models_usage=RequestUsage(prompt_tokens=0, completion_tokens=0) metadata={} created_at=datetime.datetime(2025, 8, 9, 20, 36, 40, tzinfo=datetime.timezone.utc)
----- ToolCallRequestEvent (interviewer_agent) -----
[FunctionCall(id='call_QwLyZsApX68krD85eF53LAeP', arguments={'speaker_or_records': [{'speaker': 'interviewer', 'message': 'QUESTION: How would you define success for this project?'}]}, name='call_QwLyZsApX68krD85eF53LAeP', is_error=False)
id='433c2d88-d35f-4de8-91a9-775e70664388' source='interviewer_agent' models_usage=None metadata={} created_at=datetime.datetime(2025, 8, 9, 20, 36, 40, tzinfo=datetime.timezone.utc)
id='433c2d88-d35f-4de8-91a9-775e70664388' source='interviewer_agent' models_usage=None metadata={} created_at=datetime.datetime(2025, 8, 9, 20, 36, 40, tzinfo=datetime.timezone.utc)
DEBUG:autogen_agentchat.events:id='433c2d88-d35f-4de8-91a9-775e70664388' source='interviewer_agent' models_usage=None metadata={} created_at=datetime.datetime(2025, 8, 9, 20, 36, 40, tzinfo=datetime.timezone.utc)
----- ToolCallExecutionEvent (interviewer_agent) -----
[FunctionExecutionResult(content='Saved 2 record(s) to ../conversation/interview.jsonl', name='save_conversation_to_jsonl', call_id='call_QwLyZsApX68krD85eF53LAeP', is_error=False)
----- ModelClientStreamingChunkEvent (interviewer_agent) -----
Product Overview asked=2, status=done
KPI & Success Metrics asked=1, status=pending
Scope In and Out asked=0, status=not started
Non-Functional Requirements asked=0, status=not started
User Roles & Personas asked=0, status=not started
Integrations & External Systems asked=0, status=not started
Constraints & Assumptions asked=0, status=not started
Dependencies & Risks asked=0, status=not started

QUESTION: Should additional reporting capabilities or real-time data visualization options be part of the MVP?
/run/media/jun/Jun 6T/Code/TFM_BA_GPT/venv/lib/python3.13/site-packages/autogen_agentchat/teams/_group_chat/_selector_group_chat.py:269: UserWarning: Resolved model mismatch
  response = await self._model_client.create(messages=select_speaker_messages)
Model selected the previous speaker: interviewer_agent (attempt 1)
DEBUG:autogen_agentchat:Model selected the previous speaker: interviewer_agent (attempt 1)
Model selected a valid name: planning_agent (attempt 2)
DEBUG:autogen_agentchat:Model selected a valid name: planning_agent (attempt 2)
Selected speaker: planning_agent
DEBUG:autogen_agentchat:Selected speaker: planning_agent
----- ModelClientStreamingChunkEvent (planning_agent) -----
1 user : Answer the interviewer_agent's last question about additional reporting capabilities or real-time data visualization options in the MVP.

```

Nota. Se utiliza “screenshot” ya que los textos son muy largos a lo ancho y a pegarlos completos se hace muy difícil la lectura. Para ver parte del texto ver Apéndice E

4.2.7 Caso real “AI Dialer” – Análisis funcional MAS de la traza

En esta sección examinamos la traza ejecutiva desde una perspectiva funcional, con el objetivo de mostrar cómo se coordinan internamente los agentes y de qué manera se materializan las decisiones de orquestación (selección del hablante, definición del plan y asignación de tareas). A diferencia del análisis previo centrado en métricas de eficiencia y calidad conversacional— aquí nos enfocamos en la dinámica operativa: el rol de SelectorGroupChat en la elección del planner, la formulación del plan de trabajo por parte del planner y las primeras derivaciones hacia los agentes ejecutores.

Para facilitar la lectura y mantener la trazabilidad semántica, presentamos extractos literales y mínimos de la traza, curados para evidenciar los hitos de coordinación sin sobrecargar al lector con detalles accesorios. Se conserva el contenido textual original, pero se omiten identificadores técnicos y marcas temporales que no aportan al argumento. El orden de los fragmentos respeta la secuencia cronológica de ejecución.

El recorrido comienza con la asignación inicial (SelectorGroupChat → planner) y el despiece del plan del planner; en apartados posteriores se documentan los handoffs hacia los siguientes agentes (p. ej., solicitud de resumen, validación y escritura), siguiendo el mismo criterio de presentación y evidenciando la coordinación paso a paso del sistema.

Extracto de traza - Inicio de la orquestación (SGC → Planner → Plan inicial)

1. SelectorGroupChat elige al planner

Model selected a valid name: **planning_agent** (attempt 1)

Selected speaker: **planning_agent**.

2. El planner traza el estado y el plan

Estado inicial de agentes

interviewer_agent: **not started**

summarizer_agent: **not started**

functional_spec_writer_agent: **not started**

functional_spec_checker_agent: **not started**

diagram_creator_agent: **not started**

mermaid_code_reviewer_agent: **not started.**

Desglose del plan

Breaking down the task:

1. Start by gathering product requirements...
2. Summarize all collected information.
3. Write the functional specification document.
4. Review the functional specification...
5. Create process flow diagrams...
6. Review the Mermaid code for the diagrams.

3. Primera asignación del planner

Assigning the first task:

1 **interviewer_agent** : Start gathering requirements from the user for the new product.

[ASSIGNMENT] agent=**interviewer_agent** task="Start gathering requirements..."

Confirmación de SGC de la asignación

Model selected a valid name: **interviewer_agent** (attempt 1)

Selected speaker: **interviewer_agent**.

4. **Evidencia de un ciclo posterior: planner vuelve a tomar la palabra y deriva al summarizer**

Model selected a valid name: **planning_agent** (attempt 1) → Selected speaker: **planning_agent**.

1 **summarizer_agent** : Summarize gathered requirements...

[ASSIGNMENT] agent=**summarizer_agent** task="Summarize gathered requirements..."

Luego SGC selecciona a **summarizer_agent**.

Extracto de traza — Solicitud de “APPROVE SUMMARY?” y orquestación (SGC → Summarizer → Writer → Checker)

1. SelectorGroupChat selecciona al summarizer

Model selected a valid name: **summarizer_agent** (attempt 1)

Selected speaker: **summarizer_agent**

2. Planner asigna el resumen

[ASSIGNMENT] agent=**summarizer_agent** task="Summarize gathered requirements ...

prepare the summary for the system proposal."

3. El summarizer solicita aprobación (“APPROVE SUMMARY?”)

“Would you like to add or clarify any information, or do you **APPROVE the summary?**”

4. SelectorGroupChat selecciona al writer

Model selected a valid name: **functional_spec_writer_agent**

Selected speaker: **functional_spec_writer_agent**

5. Planner deriva al writer

1 **functional_spec_writer_agent**: Finalize the Functional Specification Document...

[ASSIGNMENT] agent=**functional_spec_writer_agent** task="Finalize... using the completed

Use Cases and Glossary sections."

6. SelectorGroupChat selecciona al checker

Model selected a valid name: **functional_spec_checker_agent**

Selected speaker: **functional_spec_checker_agent**

7. Planner deriva al checker

1 functional_spec_checker_agent: Review the finalized Functional Specification Document...

[ASSIGNMENT] agent=**functional_spec_checker_agent** task="Review... for completeness and accuracy."

Extracto de traza — Diagramación, revisión y cierre

1. SelectorGroupChat selecciona al Diagram Creator

Model selected a valid name: **diagram_creator_agent** (attempt 1)

Selected speaker: **diagram_creator_agent**

2. Diagram Creator lee el FSD y actualiza Canvas

read_md_file_tool → **functional_specification.md**

canvas.update_file_tool → **Canvas_Memory_canvas_diagram.md**

3. SelectorGroupChat selecciona al Mermaid Code Reviewer

Model selected a valid name: **mermaid_code_reviewer_agent**

Selected speaker: **mermaid_code_reviewer_agent**

4. Reviewer itera sobre Canvas y decide aprobar

canvas.update_file_tool → revisión/ajustes sobre **canvas_diagram.md**

“Diagram **approved.**”

5. Commit final del diagrama en .mmd

write_mermaid_to_file_tool → **conversation/process_diagram.mmd**

6. Terminate: mover/limpiar/renderizar/persistir

terminate_agent → move_file → **experience/<SESSION_ID>/**

clean_mermaid → render_mermaid_png → format_and_export_pdf

save_experience_to_chromadb

Extracto de traza — Terminador: empaqueta artefactos, normaliza Mermaid, renderiza y persiste

1. Disparo del pipeline de cierre

El agente de cierre inicia explícitamente el flujo final:

“Execute the termination pipeline”.

2. Consolidación y estructura de sesión

Crea la carpeta de experiencia por session_id, asegurando trazabilidad y reanudación futura:

“Successfully created directory .../experience/sess-20231011162620”.

3. Recolección de artefactos (movido atómico)

Traslada los artefactos clave desde conversation/ a la sesión persistente:

- interview.jsonl (registro Q&A de la entrevista)
- functional_specification.md (FSD)
- process_diagram.mmd (diagrama Mermaid)

“Successfully moved .../interview.jsonl .../functional_specification.md

.../process_diagram.mmd”.

4. Higiene sintáctica de Mermaid (pre-render)

Ejecuta limpieza/normalización del código Mermaid previo al render:

clean_mermaid → detected_type: "flowchart", issue DEDENTED (indentación común removida).

5. Render del diagrama

Genera el artefacto gráfico para inclusión documental:

“PNG written to .../**process_diagram.png**”.

6. Formateo y export a PDF

Integra el FSD formateado + diagrama y produce el entregable final:

- Salidas: functional_specification.formatted.md y deliverable.pdf
- Métricas extraídas: business_requirements: 5, functional_requirements: 7

“written_to .../**functional_specification.formatted.md** ... pdf .../**deliverable.pdf**”.

7. Verificación del entregable

Audita el PDF para control de calidad y preservación:

size: 89854, created/modified: 2025-09-06 21:38:16, permissions: 644.

8. Indexación en memoria episódica (persistencia semántica)

Indexa la sesión completa en el repositorio de experiencias (para RAG/retoma):

{ "ok": true, "session_id": "sess-20231011162620", "files_indexed": 6, "written_items": 22 }.

9. Señal de fin

Cierre explícito del workflow:

“**TERMINATE**”.

Se puede ver el Functional_Specification_Document.pdf al final del Apéndice E

4.3 Nivel de Satisfacción del Usuario

4.3.1 Muestra y alcance

Se aplicaron 21 encuestas de satisfacción (una por cada combinación experto × producto), resultantes de 7 expertos evaluando 3 productos reales: (1) AI Dialer, (2) Human Resource Management software, (3) Online Apparels Shopping Website. El panel estuvo conformado por dos Project Managers (PM), dos Business Analysts (BA), un Solution Architect (SA), un Platform Manager y un Operations Manager.

4.3.2 Instrumento (estructura y número de ítems)

Se diseñó un cuestionario semiestructurado con 17 preguntas por encuesta, compuesto por:

- 14 ítems cerrados en escala Likert de 5 puntos (1 = Muy en desacuerdo ... 5 = Muy de acuerdo).
- 3 preguntas abiertas para comentarios cualitativos.

Bloques de evaluación (y su correspondencia con la Sección 4.3)

1. Utilidad percibida (2 ítems) → “Satisfacción entre media y alta” , ahorro de tiempo.
2. Claridad de resultados (2 ítems) → Coherencia y legibilidad de documentos/diagramas.
3. Velocidad y latencia (2 ítems) → “Comentarios positivos sobre la velocidad” .
4. Grado de automatización (2 ítems) → Capacidad de orquestación sin intervención manual.
5. Alineación y trazabilidad (2 ítems) → “Coherencia y alineación con las expectativas” .
6. Calidad de artefactos (2 ítems) → Tablas/diagramas; recoge críticas sobre tablas iniciales y diagramas básicos.
7. Aplicabilidad futura (2 ítems) → Posibilidad de adopción en proyectos reales.

Cualitativo (3 preguntas abiertas):

- Fortalezas percibidas.
- Aspectos para mejorar (p. ej., tablas de requisitos, complejidad de diagramas).
- Casos de uso/ámbitos de aplicación.

4.3.3 Procedimiento de aplicación

Cada experto respondió una encuesta por producto inmediatamente después de finalizar el flujo de entrevista, síntesis y generación de artefactos (documento funcional y diagrama). El tiempo medio de cumplimentación fue de 6–8 minutos por encuesta. Se obtuvieron 21/21 respuestas (100 % de tasa de respuesta), lo que asegura cobertura plena de los tres casos.

4.3.4 Análisis

- Cuantitativo: los 14 ítems se analizaron por bloque (media/mediana y dispersión). A efectos de reporte en 4.3, se emplearon categorías agregadas coherentes con la narrativa:
 - Baja: 1,0–2,4 | Media: 2,5–3,4 | Alta: 3,5–5,0.

Esto permite sostener la descripción “valoración entre media y alta” y distinguir áreas con margen de mejora (p. ej., tablas de requisitos y diagramas complejos).
- Cualitativo: se realizó codificación temática de las 3 preguntas abiertas para extraer fortalezas (p. ej., velocidad, coherencia, ahorro de tiempo) y oportunidades de mejora (p. ej., enriquecer tablas y diagramas)

4.3.5 Trazabilidad con los productos evaluados

Los resultados se reportan por producto y de forma agregada, manteniendo la trazabilidad con los tres escenarios de prueba. En la Sección 4.3 se hace referencia a las dimensiones

evaluadas y a las conclusiones globales; los instrumentos y datos de detalle quedan documentados en los Apéndices para consulta y auditoría.

Resumen ejecutivo: Con 21 encuestas (7 expertos $\times$ 3 productos) y 17 preguntas por instrumento (14 Likert + 3 abiertas), el diseño garantiza suficiente granularidad para respaldar los resultados, manteniendo una carga razonable para los participantes y una clara alineación entre instrumento, resultados y conclusiones.

4.3.6 Conclusiones de los análisis de entrevistas y encuestas al panel experto

A partir de las 21 encuestas (7 expertos $\times$ 3 productos) y del examen de los registros JSONL de las entrevistas, se derivan las siguientes conclusiones:

1. Calidad semántica y coherencia.

Las conversaciones muestran alineación temática sostenida (preguntas pertinentes al sujeto activo y seguimiento focalizado de NEEDS_INFO), con formulaciones claras que favorecen respuestas accionables. El handoff al summarizer_agent mantiene la fidelidad del contenido y facilita la validación con el usuario.

2. Ritmo conversacional y experiencia de uso.

El ciclo pregunta $\rightarrow$ respuesta $\rightarrow$ persistencia se desarrolla con latencia baja y tiempos naturales para entrevistas guiadas (flujo “near-realtime”). Los expertos perciben agilidad y reducción de esfuerzo en comparación con sesiones tradicionales.

3. Utilidad percibida y trazabilidad.

Se valora positivamente la utilidad práctica del sistema y su trazabilidad extremo a extremo: desde la entrevista hasta el documento funcional, pasando por el control de estados (READY/NEEDS_INFO) y el registro pareado de Q&A en JSONL.

4. Calidad de artefactos y presentación.

El documento funcional y los diagramas satisfacen expectativas para una primera versión. La mejora reciente en diagramas y maquetación PDF eleva la calidad percibida del entregable.

Persisten oportunidades de mejora en:

- Tablas de requisitos (normalización de formato, consistencia de campos).
- Diagramas (mayor riqueza visual cuando el proceso lo requiera).

5. Adopción y escalabilidad.

El panel identifica viabilidad funcional en contextos reales y flexibilidad para integrar distintos flujos (ASR/TTS, Realtime, carga de scripts). Se considera factible su adopción incremental en equipos de análisis y PMO, con entrenamiento ligero para el uso de la interfaz.

6. Límites y riesgos observados.

La calidad final depende de la precisión del insumo (transcripts y respuestas del usuario). En escenarios muy complejos, el posprocesado de tablas y la revisión de diagramas siguen siendo recomendables. El ramal telefónico (SIP/Twilio) añade complejidad operativa, aunque sin comprometer la experiencia.

Síntesis: El sistema ofrece beneficio neto en velocidad, coherencia y trazabilidad, con satisfacción global entre media y alta en el panel experto. Las líneas de mejora priorizadas tablas de requisitos y expresividad de diagramas quedan documentadas y se acompañan de acciones de refinamiento continuo. Los instrumentos y resultados detallados (cuestionarios e insumos de análisis) se incluyen en los Apéndices para referencia y auditoría.

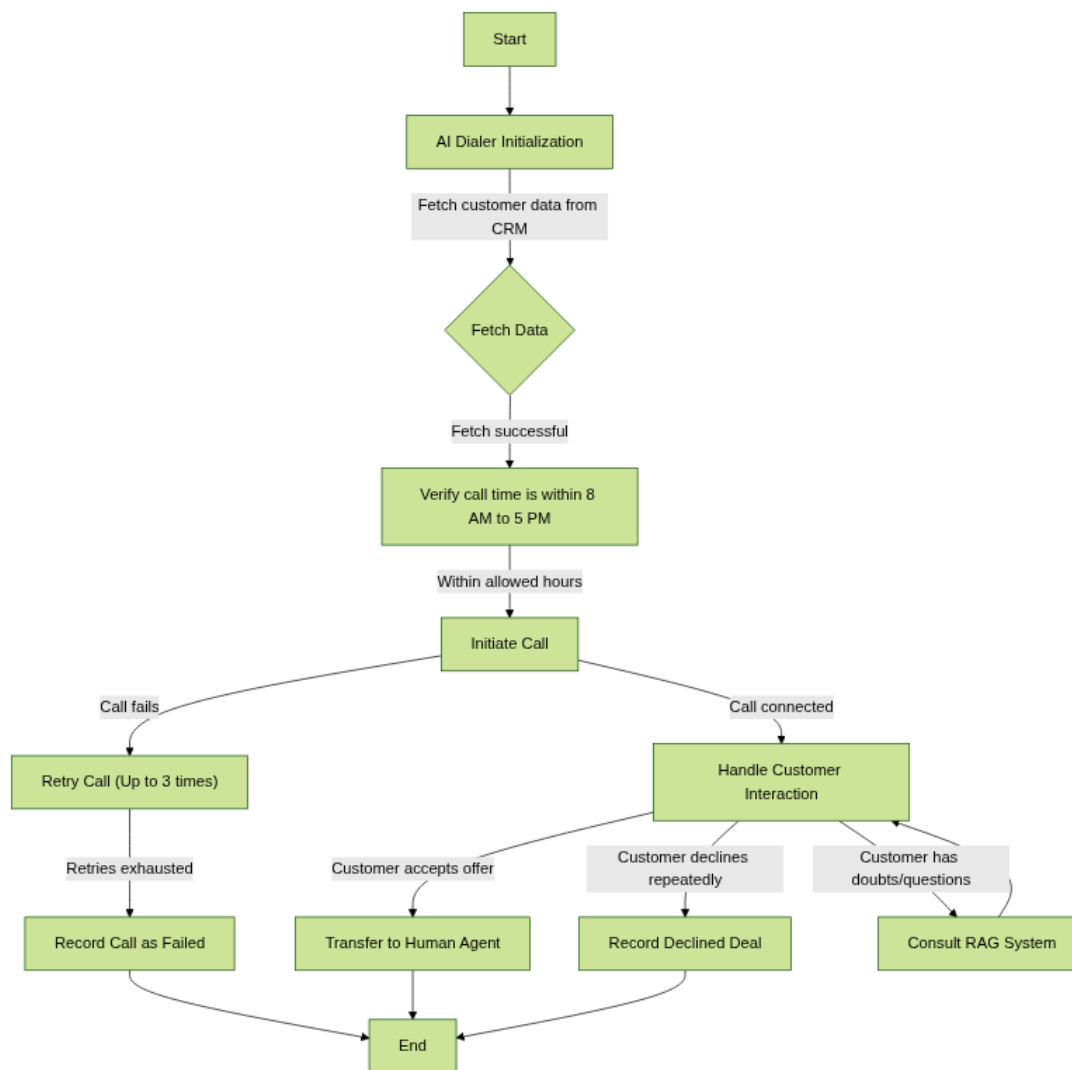
4.4 Diagrama y Gráfico: Calidad Funcional

La generación automática de diagramas del flujo de procesos se calificó como de calidad media, aunque se valoró notablemente la rapidez y coherencia general.

Actualización (agosto de 2025). Las mejoras en la claridad de los diagramas y en el formato del documento PDF elevan sustancialmente la calidad percibida, reforzando la presentación profesional del entregable.

Figura 31

Diagrama de proceso generado por Business Analyst GPT para AI Dialer solo con la información de la entrevista



Nota: Ver código Mermaid del diagrama en Apéndice E

4.5 Ahorro Global y Beneficios

- Se estima que BusinessAnalystGPT redujo en al menos 70 % el tiempo total en actividades de reunión y documentación preliminar.
- El enfoque liberó entre 8 y 15 horas semanales del equipo, permitiendo dedicar más tiempo a tareas de análisis profundo o diseño estratégico.

4.6 Conclusión de Resultados

De forma cualitativa, el sistema ha logrado:

- Revolucionar el tiempo de entrega, pasando de semanas a horas o pocos días.
- Presentar una calidad semántica robusta y altamente apreciada, especialmente con GPT-4o.
- Generar un alto nivel de satisfacción, con leves críticos centrándose en aspectos estructurales que pueden mejorarse manualmente sin costo significativo en tiempo.
- Ahorrar tiempo considerable en coordinación, evitando reuniones redundantes y facilitando la integración de interacciones con usuarios.

Estos hallazgos respaldan la viabilidad de BusinessAnalystGPT como herramienta asistida por IA para llevar a cabo tareas de análisis de negocio y documentación técnica, ofreciendo un balance entre automatización eficiente, adaptabilidad humana y calidad semántica.

5. Propuesta de Negocio

5.1 Resumen Ejecutivo

BusinessAnalystGPT es una solución de inteligencia artificial basada en agentes colaborativos que automatiza una de las tareas más críticas y costosas en cualquier proyecto digital: la recolección de requisitos funcionales y la generación de documentación técnica inicial. Mediante el uso de modelos LLM avanzados (como GPT-4), RAG, memoria contextual y orquestación multiagente, la herramienta simula el trabajo conjunto de un “business analyst”, un documentador, un verificador y un diagramador. El resultado: especificaciones funcionales de alta calidad generadas en horas en lugar de semanas, liberando tiempo de Analistas, reduciendo errores de comunicación y acelerando drásticamente el inicio de proyectos tecnológicos.

5.2 Problema para Resolver

- El proceso tradicional de análisis de negocio requiere reuniones repetitivas, entrevistas, validaciones manuales y redacción técnica.
- El tiempo promedio para generar una documentación inicial ronda entre 4 y 8 semanas, y varía según la complejidad del problema y la disponibilidad de los *stakeholders* (GOV.UK, 2021; Scottish Government, s. f.).
- El costo de errores o ambigüedad en los requisitos puede derivar en sobrecostos de desarrollo de entre 21 % y 43 % del esfuerzo de desarrollo (ver Apéndice A).
- No existe un sistema accesible que automatice este proceso de forma colaborativa, trazable y semánticamente precisa.

5.3 Solución Propuesta

BusinessAnalystGPT automatiza el flujo completo:

- 1. Entrevista con el usuario final (vía chat o audio).
- 2. Extracción de requisitos utilizando técnicas de NLP y RAG.
- 3. Síntesis del contenido y validación con el usuario (human-in-the-loop).
- 4. Generación del Functional Specification Document.
- 5. Creación automática de diagramas de flujo y validación estructural.
- 6. Trazabilidad total de decisiones vía OpenTelemetry y análisis en Jaeger.

Además, el sistema puede usar transcripciones de reuniones, eliminando la necesidad de entrevistas activas.

5.4 Propuesta de Valor

Tabla 4
Beneficio / Valor agregado

Beneficio	Valor agregado
Ahorro de tiempo	De semanas a horas en la entrega de especificaciones
Mejora de calidad	Reducción de ambigüedad y errores de interpretación
Costos reducidos	Menos reuniones, menor esfuerzo manual
Integración simple	Compatible con Slack, Teams, Jira, Salesforce vía MCP
Adaptabilidad	Personalización por industria o dominio con LoRA o RLHF
Escalabilidad	Agentes modulares para dominios múltiples

5.5 Modelo de Negocio

SaaS B2B (Business-to-Business) con niveles de suscripción:

- Starter (Gratis): 1 proyectos/mes, sin colaboración externa.
- Pro (€249/mes): 10 proyectos, integración con Slack/Teams (ver Apéndice C).
- Enterprise (custom): ilimitado, con soporte para Salesforce/Jira/API privada.

Servicios adicionales:

- Entrenamiento de modelos internos (LoRA/RLHF)
- Personalización para verticales (banca, salud, industria)
- Integración a medida (MCP, APIs empresariales)

Tabla 5
Planes y precios

Plan	Precio (€/mes)	Incluye	Límite	Excedente
Starter	Gratis	1 proyectos/mes	1 proyectos	—
Pro	249 € (199 €/mes con pago anual)	Hasta 10 proyectos/mes integraciones Slack/Teams	10 proyectos	20 € por proyecto adicional
Enterprise	A medida	Proyectos ilimitados soporte para Salesforce/Jira/API privada controles avanzados de seguridad	—	—

5.6 Mercado Objetivo

- Empresas de software (consultoras, startups, fábricas de software)
- Departamentos IT internos en organizaciones medianas y grandes
- Equipos de producto o innovación que gestionan múltiples soluciones digitales

Tamaño estimado (LATAM + Europa):

- $\approx 650\,000$ organizaciones con necesidades recurrentes de análisis funcional
- $\approx 2\text{-}5\text{M}$ de proyectos al año que inician con reuniones de relevamiento de requisitos (ver Apéndice B).

5.7 Competencia

Tabla 6
Competidor / Diferenciador

Competidor	Diferenciador BusinessAnalystGPT
ChatDev / MetaGPT	Más orientados a desarrollo técnico, no análisis
Confluence AI	No genera documentos técnicos completos
Jira AI / Notion AI	No ofrece generación estructurada ni validación multiagente

5.8 Estrategia de Implementación

- Fase 1 – Minimum Viable Product (MVP) validado (ya realizado)
- Fase 2 – Beta cerrada con early adopters (1–3 meses)
- Fase 3 – Lanzamiento SaaS (versión web e integración API)
- Fase 4 – Alianzas con consultoras y partners tecnológicos

Roadmap tecnológico (12 meses)

- Sistema multiagente con AutoGen
- RAG y memoria contextual
- Integración con Slack, Teams y WhatsApp vía MCP
- Agente multimodal para entrevistas por voz
- Personalización por dominio (usando LoRA + RLHF)
- GraphRAG para generar documentación técnica con memoria visual

Equipo fundador

- CTO / Arquitecto: Especialista en IA generativa, NLP y AutoGen
- PM/BA Senior: Experto en documentación funcional y herramientas de análisis
- Fullstack Dev: Desarrollo e integración de interfaces web y APIs
- Experto en ventas B2B: Comercialización en sectores IT, financiero y salud

Requerimientos iniciales

Inversión inicial estimada (solo OPEX de infra + observabilidad)

Tabla 7

Inversión inicial estimada (sólo infraestructura y observabilidad)

Escenario	Horizonte	Carga supuesta	Importe (con contingencia)
Piloto	6 meses	20 proyectos/mes	≈ 1,54 k – 2,55 k €
Go-to-market “lean”	12 meses	100 proyectos/mes	≈ 7,72 k – 17,44 k €

Notas Cómo se calculó (ver Apéndice C): (Overhead plataforma 131–166 €/mes, Tabla 8) + (variable por proyecto 4,12–11,16 €, Tabla 9) × volumen mensual; se aplicó 20 % (piloto) y 15 % (GTM) de contingencia.

6. Conclusiones

A lo largo del desarrollo de este Trabajo de Fin de Máster, logré cumplir satisfactoriamente con el objetivo personal de explorar, aplicar y comprender en profundidad las últimas tendencias en inteligencia artificial basada en agentes, particularmente en entornos colaborativos con “Large Language Models”. Este proyecto me permitió no solo adquirir conocimientos teóricos avanzados sobre arquitecturas multiagente, coordinación deliberativa y marcos como TB-CSPN y MAD, sino también experimentar de forma práctica con herramientas de vanguardia.

La elección de AutoGen como “framework” base fue especialmente acertada. Su enfoque modular, experimental y altamente extensible me permitió avanzar de manera progresiva desde flujos simples hacia configuraciones más sofisticadas, incorporando dinámicas de debate, planificación distribuida, RAG y memoria contextual. Comencé a trabajar con la versión 0.4 de AutoGen, y fui integrando nuevas funcionalidades conforme el “framework” evolucionaba, hasta llegar a la versión 0.6.2 actual. Esta evolución continua no solo enriqueció el proyecto, sino que me permitió estar inmerso en el proceso activo de innovación de la comunidad.

Tuve la oportunidad de interactuar directamente con los desarrolladores del “framework” y otros usuarios avanzados, participando en foros, reportando bugs, discutiendo soluciones y proponiendo mejoras. De manera particular, propuse y documenté un “*workaround*” para un problema específico que posteriormente fue corregido en una versión oficial, y también sugerí una funcionalidad que fue incorporada como nuevo “feature” en el “roadmap”. Esta experiencia reforzó mi comprensión del ecosistema “open source” y de la importancia del “feedback” técnico en ciclos ágiles de desarrollo.

En cuanto al objetivo práctico del proyecto, logré desarrollar un MVP (“Minimum Viable Product”) funcional para el asistente BusinessAnalystGPT, con un rendimiento aceptable y capacidades reales de automatización de recolección de requisitos y generación de documentación funcional. Asimismo, fue posible delinear un “roadmap” realista hacia una versión de producción, incluyendo mejoras en usabilidad, robustez, integración con plataformas corporativas (vía MCP) y soporte para modelos locales y en la nube.

En resumen, este trabajo no solo me permitió alcanzar mis metas académicas, sino también consolidar habilidades técnicas clave en diseño de sistemas con LLMs, arquitectura multiagente, ingeniería de “prompts”, trazabilidad, evaluación y despliegue de soluciones basadas en IA. La experiencia ha sido profundamente enriquecedora tanto en lo profesional como en lo personal, y marca un punto de partida sólido para desarrollos futuros en el campo de la inteligencia artificial aplicada a procesos empresariales.

7. Limitaciones y Futuras Líneas de Investigación

A pesar de los resultados positivos obtenidos en esta fase de validación, el desarrollo de BusinessAnalystGPT presenta varias limitaciones que condicionaron tanto la profundidad de la evaluación como la amplitud de las funcionalidades exploradas. Estas restricciones, principalmente de tipo temporal, técnico y operativo, señalan además rutas prometedoras para líneas futuras de investigación y desarrollo.

7.1 Limitaciones

Limitaciones de tiempo y disponibilidad humana:

Por tratarse de un sistema que requiere interacción activa con usuarios reales, no fue posible realizar un número mayor de entrevistas ni generar más ejemplos funcionales en contextos reales. Del mismo modo, las encuestas de satisfacción sólo pudieron ser aplicadas a un grupo reducido de expertos, lo que limita la generalización de los resultados.

Limitaciones técnicas en benchmarking y arquitectura avanzada:

Aunque AutoGen permite la implementación de flujos basados en grafos a través de GraphFlow, no se realizaron pruebas comparativas respecto al rendimiento entre coordinación secuencial, deliberativa y basada en grafos. Tampoco fue posible evaluar arquitecturas de memoria más sofisticadas como GraphRAG, especialmente útiles para representar estructuras como diagramas complejos o relaciones semánticas de alto nivel.

Limitaciones de evaluación estandarizada:

No se pudo implementar un “benchmark” avanzado como AutoGenBench, lo cual habría permitido realizar una evaluación comparativa más robusta y replicable frente a otros sistemas de agentes LLM. Esta omisión restringe la capacidad para cuantificar el desempeño bajo estándares establecidos.

Limitaciones de integración externa:

Aunque el sistema está diseñado con soporte para MCP (“Model Context Protocol”), no se logró integrar BusinessAnalystGPT con plataformas empresariales como Slack, Microsoft Teams o WhatsApp, debido a la complejidad técnica y restricciones de tiempo. Tampoco se exploró la conexión con sistemas corporativos como Jira, Salesforce o ServiceNow, por requerir acceso a APIs comerciales o licencias empresariales.

Restricciones de hardware y cómputo local. Esto también impidió explorar técnicas de fine-tuning o el uso de LoRA adapters para especializar modelos por tipo de agente.

Acceso limitado a software pago o licencias comerciales:

Algunas integraciones y funcionalidades avanzadas, por ejemplo, conectores para sistemas de gestión empresarial, plataformas de ventas o herramientas ITSM no pudieron realizarse por falta de acceso a cuentas de pago o entornos corporativos reales.

7.2 Futuras Líneas de Investigación y Mejoras

En función de las limitaciones anteriores, se identifican varias líneas de desarrollo futuro: Ampliar el número y diversidad de casos de uso reales, incluyendo verticales específicas (salud, finanzas, e-commerce).

Explorar nuevas modalidades de interacción, como entrevistas por- video, utilizando modelos multimodales.

Implementar flujos de coordinación basados en grafos y semántica profunda, como GraphFlow y GraphRAG.

Evaluar el sistema con herramientas estandarizadas como AgentBench y otras métricas de robustez.

Mejorar la integración con sistemas empresariales reales a través de conectores MCP o “plugins” específicos.

Aplicar técnicas de especialización como fine-tuning, LoRA o agentes con modelos dedicados por dominio.

Soporte para utilizar transcripciones de llamadas provenientes de plataformas como Microsoft Teams, Zoom o Google Meet como insumo directo para la recolección de requisitos.

Estas mejoras no solo aumentarían el valor práctico de BusinessAnalystGPT, sino que contribuirían a validar nuevos modelos de interacción humano-IA en entornos empresariales reales.

Actualización (agosto de 2025). A partir de la experiencia acumulada, se constata que numerosas restricciones han sido efectivamente superadas e implementadas; pese a ello, subsisten desafíos puntuales y la aparición sostenida de nuevas soluciones obliga a mantener una estrategia de revisión y adopción incremental.

Referencias

Andrés, D. (2024, December 15). DIY #14 – Step-by-step implementation of a ReAct Agent in LangGraph. *Machine Learning Pills*.

<https://mlpills.substack.com/p/diy-14-step-by-step-implementation>

Anthropic. (2024). *Model Context Protocol (MCP): A secure standard for tool integration in LLM agents*.

<https://www.anthropic.com/news/model-context-protocol>

Borghoff, U. M., Bottoni, P., & Pareschi, R. (2025). An organizational theory for multi-agent interactions integrating human agents, LLMs, and specialized AI. *Discov Computing*, 28, 138.

<https://doi.org/10.1007/s10791-025-09667-2>

Chen, T., & Rickman, C. (2025, mayo 27). Semantic Kernel: Multi-agent orchestration. *Microsoft Dev Blogs*.

<https://devblogs.microsoft.com/semantic-kernel/semantic-kernel-multi-agent-orchestration/>

Cui, Y., Yao, L., Li, Z., Li, Y., Ding, B., & Zhou, X. (2025). Efficient leave-one-out approximation in LLM multi-agent debate based on introspection. *arXiv*.

<https://doi.org/10.48550/arXiv.2505.22192>

Davis, A. M., & Leffingwell, D. (2003). *Using requirements management to speed delivery of higher quality applications*. IBM Rational Software.

<https://public.dhe.ibm.com/software/rational/web/whitepapers/2003/696wp.pdf>

Department for Business & Trade. (2023, 5 de octubre). *Business population estimates for the UK and regions 2023: Statistical release*. GOV.UK.

<https://www.gov.uk/government/statistics/business-population-estimates-2023/business-population-estimates-for-the-uk-and-regions-2023-statistical-release>

DigitalOcean. (2025). *Budget friendly cloud server pricing*.

<https://www.digitalocean.com/pricing>

EURES. (2025). *Labour Market Information: Switzerland*. European Union.

https://eures.europa.eu/living-and-working/labour-market-information-europe/labour-market-information-switzerland_en

Eurostat. (2024). *Structural business statistics overview (Statistics Explained)*.

https://ec.europa.eu/eurostat/statistics-explained/index.php/Structural_business_statistics_overview

Eurostat. (2024, 25 de octubre). Micro & small businesses make up 99% of enterprises in the EU. *Eurostat News*.

<https://ec.europa.eu/eurostat/web/products-eurostat-news/w/ddn-20241025-1>

European Central Bank. (2025, 2 de septiembre). *Euro foreign exchange reference rates*.

<https://www.ecb.europa.eu/stats/eurofxref/>

Federal Statistical Office (FSO). (2024, 22 de agosto). *Structural Business Statistics 2022 – Record employment growth in secondary sector*. admin.ch.

<https://www.news.admin.ch/en/nsb?id=102098>

Ferber, J. (1999). *Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence*. Addison-Wesley.

Gitnux. (2025). *Software Industry Statistics: Market Data Report 2025*.

<https://gitnux.org/software-industry-statistics/>

GOV.UK. (2021, 21 de junio). *How the discovery phase works*. Government Digital Service.

<https://www.gov.uk/service-manual/agile-delivery/how-the-discovery-phase-works>

Griciūnas, A. (2024, April 30). Simple way to explain memory in AI agents. *SwirlAI Newsletter*.

<https://www.newsletter.swirlai.com/p/simple-way-to-explain-memory-in-ai>

Gruber, T. R. (1993). A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5(2), 199–220.

Gupta, D. (2025, May 9). MCP explained: The protocol revolutionizing AI capabilities.

<https://guptadeepak.com/mcp-a-comprehensive-guide-to-extending-ai-capabilities>

Hexgrad. (2024). *Kokoro-82M* [Model card]. Hugging Face.

<https://huggingface.co/hexgrad/Kokoro-82M>

Hong, S., Zhuge, M., Chen, J., Zheng, X., Cheng, Y., Zhang, C., Wang, J., Wang, Z., Yau, S. K. S., Lin, Z., Zhou, L., Ran, C., Xiao, L., Wu, C., & Schmidhuber, J. (2024). MetaGPT: Meta programming for a multi-agent collaborative framework (Versión 7) [Preprint]. *arXiv*.

<https://doi.org/10.48550/arXiv.2308.00352>

Jaeger Project. (2025). *Jaeger: Open source, distributed tracing platform*.

<https://www.jaegertracing.io/>

Jensen, K., & Kristensen, L. M. (2009). *Coloured Petri Nets: Modelling and Validation of Concurrent Systems*. Springer.

Kokoro TTS. (s. f.). *Kokoro TTS: Advanced text-to-speech*.

<https://kokorotts.net/>

LeewayHertz. (2024). *Agentic RAG: What it is, its types, applications and implementation*.

<https://www.leewayhertz.com/agentic-rag/>

Leffingwell, D. (2003). *Calculating your return on investment from more effective requirements management*. IBM Rational Software.

<https://public.dhe.ibm.com/software/rational/web/whitepapers/2003/roi1.pdf>

Luo, J., Zhang, W., Yuan, Y., Zhao, Y., Yang, J., Gu, Y., Wu, B., Chen, B., Qiao, Z., Long, Q., Tu, R., Luo, X., Ju, W., Xiao, Z., Wang, Y., Xiao, M., Liu, C., Yuan, J., Zhang, S., Jin, Y., Zhang, F., Wu, X., Zhao, H., Tao, D., Yu, P. S., & Zhang, M. (2025, March 27). Large language

model agent: A survey on methodology, applications and challenges (Version 1) [Preprint].
arXiv.

<https://doi.org/10.48550/arXiv.2503.21460>

Microsoft. (2025). *How to use the GPT-4o Realtime API for speech and audio (Preview)*.

Microsoft Learn.

<https://learn.microsoft.com/en-us/azure/ai-foundry/openai/how-to/realtime-audio>

Murata, T. (1989). Petri nets: Properties, analysis and applications.

Proceedings of the IEEE, 77, 541–580.

Murphy, M. (2014, 12 de junio). Businesses take ‘average 5 months’ to deliver IT projects.

Computerworld.

<https://www.computerworld.com/article/1695625/businesses-take-average-5-months-to-deliver-it-projects.html>

OECD. (s. f.). *Enterprises by business size*.

<https://www.oecd.org/en/data/indicators/enterprises-by-business-size.html>

OpenAI. (2022). *Whisper* [Software]. GitHub.

<https://github.com/openai/whisper>

OpenAI. (2023, 1 de marzo). Introducing ChatGPT and Whisper APIs.

<https://openai.com/index/introducing-chatgpt-and-whisper-apis/>

OpenAI. (2024). *Whisper large-v3* [Model card]. Hugging Face.

<https://huggingface.co/openai/whisper-large-v3>

OpenAI. (2024, 25 de enero). New embedding models and API updates.

<https://openai.com/index/new-embedding-models-and-api-updates/>

OpenAI. (2024a). Introducing the Realtime API.

<https://openai.com/index/introducing-the-realtime-api/>

OpenAI. (2025). *API pricing*.

<https://openai.com/api/pricing/>

OpenAI. (s. f.-a). *Realtime API — Guide*. OpenAI Platform Docs.

<https://platform.openai.com/docs/guides/realtime>

OpenAI. (s. f.-b). *Realtime API — Sending events (WebRTC/WebSocket)*. OpenAI Platform Docs.

<https://platform.openai.com/docs/guides/realtime/sending-events>

OpenAI. (s. f.-c). *gpt-4o-mini-realtime-preview — Model reference*. OpenAI Platform Docs.

<https://platform.openai.com/docs/models/gpt-4o-mini-realtime>

OpenTelemetry Authors. (2025). *Documentation*.

<https://opentelemetry.io/docs/>

Petri, C. A. (1962). *Kommunikation mit Automaten*.

Schriften des Instituts für Instrumentelle Mathematik, Bonn.

Pinecone Systems, Inc. (2025). *Pricing*.

<https://www.pinecone.io/pricing/>

Project.co. (2024). *Project Management Statistics 2024*.

<https://project.co/project-management-statistics/>

Programa de las Naciones Unidas para el Desarrollo (PNUD). (2025). *Resilient Micro, Small, and Medium Enterprises for and by Prosperity*.

<https://www.undp.org/latin-america/blog/resilient-micro-small-and-medium-enterprises-and-prosperity>

Qian, C., Liu, W., Liu, H., Chen, N., Dang, Y., Li, J., Yang, C., Chen, W., Su, Y., Cong, X., Xu, J., Li, D., Liu, Z., & Sun, M. (2024). ChatDev: Communicative agents for software development (Versión 5) [Preprint]. *arXiv*.

<https://arxiv.org/abs/2307.07924>

Radford, A., Kim, J. W., Xu, T., Brockman, G., McLeavey, C., & Sutskever, I. (2023). Robust speech recognition via large-scale weak supervision. *Proceedings of the 40th International Conference on Machine Learning (ICML 2023), Proceedings of Machine Learning Research*, 202, 28492–28518.

<https://proceedings.mlr.press/v202/radford23a.html>

Russell, S., & Norvig, P. (2020). *Artificial Intelligence: A Modern Approach* (4th ed.). Pearson.

Saran, A. (2025, January 22). Guide to understanding agentic AI frameworks. *Polestar Analytics*.

<https://www.polestarllp.com/blog/guide-to-understand-agentic-ai-frameworks>

Scottish Government. (s. f.). *Discovery*. Digital Scotland Service Manual.

<https://servicemanual.gov.scot/discovery>

Acrónimos

LLM: Large Language Model

SLM: Small Language Model

RAG: Retrieval-Augmented Generation

TB-CSPN: Topic-Based Conditional Structured Process Networks

MCP: Model Context Protocol

MAD: Multi-Agent Debate

MADR: Multi-Agent Debate with Reasoning

RLHF: Reinforcement Learning from Human Feedback

LoRA: Low-Rank Adaptation

NLP: Natural Language Processing

API: Application Programming Interface

TFM: Trabajo de Fin de Máster

BA: Business Analyst

PM: Project Manager

SA: Solution Architect

B2B: Business to Business

O&M: Operation and Maintenance

UI: User Interface

GPT: Generative Pre-trained Transformer

GTP4o: GPT-4 Omni

MVP: Minimum Viable Product

IP: Intellectual Property

ITSM: IT Service Management

TFM: Trabajo de Fin de Master

P2P: Peer to Peer (posible si se menciona)

OCR: Optical Character Recognition (si se implementa en transcripción)

BPMN: Business Process Model and Notation

CRM: Customer Relationship Management

GenAI: Generative Artificial Intelligence

IA: Inteligencia Artificial

IDE: Integrated Development Environment

LM: Language Model

ML: Machine Learning

SaaS: Software as a Service

SIP: Session Initiation Protocol

YAML: YAML Ain't Markup Language

Apéndice

Apéndice A

Cálculo de los Sobrecostos por Errores de Requisitos

Esta estimación se deriva de (a) evidencia que sitúa el rework entre 30 % y 40 % del costo total en la mayoría de las organizaciones y que, en proyectos grandes, puede acercarse a 50 % (Davis & Leffingwell, 2003; Leffingwell, 2003;), y (b) el hallazgo de que 70 %–85 % del rework proviene de errores de requisitos (Davis & Leffingwell, 2003; Leffingwell, 2003). En términos numéricos: $30 \% \times 70 \% = 21 \%$ (límite inferior) y $50 \% \times 85 \% \approx 43 \%$ (límite superior).

Apéndice B

Estimación del Tamaño del Mercado y Proyectos Anuales

Tamaño estimado (LATAM + Europa):

- $\approx 650\,000$ (rango: 620 000–680 000) organizaciones con necesidades recurrentes de análisis funcional (proxy: empresas de ≥ 50 empleados). Estimación construida sumando: Europa (UE27+Reino Unido+Suiza) $\approx 380\,000$ (UE27: $\sim 323\,000$ con ≥ 50 empleados; RU: 36 900 medianas y 8 000 grandes; Suiza: $\sim 10\,200$ medianas y 1 351 grandes) y LATAM (17 países) $\approx 271\,000$ (206 400 medianas + $\sim 64\,800$ grandes) (Eurostat, 2024; Eurostat, 2024; Department for Business & Trade [UK], 2023; EURES/FSO, 2024–2025; SELA, 2024; PNUD, 2025).
- $\approx 1,9$ – $5,3$ millones de proyectos/año que inician con reuniones de relevamiento de requisitos en LATAM+Europa. Cálculo: (i) supusimos que una organización mediana inicia 4–10 proyectos/año y una grande 12–36 (consistente con que 59 % de los PM gestionan 2–5 proyectos en paralelo y que un proyecto TI típico dura ~ 5 –9 meses); (ii) aplicamos que ≈ 52 % de los proyectos generan un documento de alcance/brief en la fase de planificación (como proxy de una reunión formal de requisitos). Aplicando (i) y (ii) sobre el universo estimado de organizaciones medianas y grandes en LATAM+Europa resulta el intervalo reportado (Wellingtone, 2021; Project.co, 2024; Murphy, 2014; Gitnux, 2025).

Notas metodológicas

Definición de “organización con necesidad recurrente de BA”. Usamos como proxy las empresas con ≥ 50 empleados (UE/OECD: “medianas 50–249; grandes ≥ 250 ”), dado que son las que más tienden a ejecutar portafolios de proyectos digitales de forma continua (OECD, s. f.).

Cobertura geográfica. “Europa” se aproxima como UE27+RU+Suiza (la mayor parte del tejido empresarial europeo); LATAM usa el recuento de 17 países informado por SELA/ECLAC. Ajustamos a rangos para absorber diferencias de fuente y año.

Proxy de “relevamiento de requisitos”. Usamos la presencia de documento de alcance en la fase de planificación (52 % de los proyectos) como indicador de que existe una ceremonia formal de descubrimiento/requisitos (Wellington, 2021).

Intensidad de proyectos por organización. Se triangula con (a) concurrencia típica de proyectos por PM (2–5 simultáneos) y (b) duración media de proyectos TI (~5–9 meses), lo que se traduce en un flujo anual razonable de inicios/cierres por organización dentro de los rangos usados (Project.co, 2024; Murphy, 2014; Gitnux, 2025).

Apéndice C

Justificación de Precio Pro

Coste variable medio por proyecto 7,64 € (ver Tabla 5.5.2); a 10 proyectos/mes → ≈ 76 € en COGS variables. Infraestructura compartida (Tabla 5.5.3) ≈ 131–166 €/mes para toda la plataforma (se prorratea entre clientes). Precios unitarios de modelos (OpenAI, 2025) y vector DB (Pinecone Systems, Inc., 2025) según tarifas públicas.

Supuestos: dos perfiles de uso.

Tabla 8
Coste variable por proyecto (en €) Perfil “ligero” (entrevista 30 min; embeddings 50 k tokens; RAG pequeño)

Componente (tarifa)	Consumo	Subtotal
GPT-4o mini (0,60 US\$/M input; 2,40 US\$/M output)	300 k in + 100 k out	0,36 €
GPT-5 (1,25 US\$/M in; 10 US\$/M out)	100 k in + 30 k out	0,36 €
Whisper (STT) (0,006 US\$/min)	30 min	0,15 €
Embeddings text-embedding-3-large (0,00013 US\$/1 k)	50 k	0,006 €
Pinecone serverless (lecturas 16 US\$/M; escrituras 4 US\$/M almacenamiento 0,33 US\$/GB/mes)	0,2 M RU + 0,1 M WU + 0,5 GB	3,24 €
Total por proyecto (ligero)		≈ 4,12 €

Componente (tarifa)	Consumo	Subtotal
GPT-4o mini (0,60 US\$/M <i>input</i> ; 2,40 US\$/M <i>output</i>)	300 k in + 100 k out	0,36 €
GPT-5 (1,25 US\$/M in; 10 US\$/M out)	100 k in + 30 k out	0,36 €
Whisper (STT) (0,006 US\$/min)	30 min	0,15 €
Embeddings text-embedding-3-large (0,00013 US\$/1 k)	50 k	0,006 €
Pinecone serverless (lecturas 16 US\$/M; escrituras 4 US\$/M almacenamiento 0,33 US\$/GB/mes)	0,2 M RU + 0,1 M WU + 0,5 GB	3,24 €
Total por proyecto (ligero)		≈ 4,12 €

Coste variable por proyecto (en €) Perfil “intenso” (entrevista 60 min; embeddings 250 k;
RAG mediano)

Componente (tarifa)	Consumo	Subtotal
GPT-4o mini	1,5 M in + 0,5 M out	1,80 €
GPT-5	300 k in + 100 k out	1,18 €
Whisper (STT)	60 min	0,31 €
Embeddings text-embedding-3-large	250 k	0,028 €
Pinecone serverless	0,5 M RU + 0,2 M WU + 1 GB	7,86 €
Total por proyecto (intenso)		≈ 11,16 €

Notas Fuentes de tarifas: OpenAI — API pricing (GPT-5, GPT-4o mini); Whisper (0,006 US\$/min); embeddings (*text-embedding-3-large*); Pinecone — Pricing (RU/WU/GB).

Tabla 9
 “Overhead” mensual de plataforma (€/mes)

Recurso	Tarifa pública	Cantidad	Total €/mes
Netlify Pro	19 US\$/usuario/mes	2–4 <i>seats</i>	33–65 €
Supabase Pro	25 US\$/mes	1	21,47 €
Compute “Micro” (Supabase)	10 US\$/mes	1	8,59 €
Pinecone Standard (mínimo)	50 US\$/mes	1	42,93 €
Sentry Team (base)	26 US\$/mes	1	22,33 €
VM para Jaeger (DigitalOcean “Basic”)	desde 4 US\$/mes (512 MB)	1	3,43 €
Subtotal mensual plataforma			≈ 131–166 €

Notas. (i) Jaeger/OpenTelemetry son open-source (sin licencias; sólo hosting). (ii) Netlify se prorratea por equipo (2–4 seats según necesidades). (iii) El mínimo de Pinecone aplica al plan Standard; Starter puede reducir temporalmente el coste si encaja en límites. Fuentes: Netlify, Supabase, Pinecone, Sentry, DigitalOcean, Jaeger/OTel.

Apéndice D

Este apéndice está dirigido a quienes deseen profundizar en la ingeniería de “*prompts*”. Explica el proceso de “*prompting*” seguido, incluyendo los criterios de diseño, las iteraciones realizadas, las decisiones de “*guardrails*” (roles, “*tracking*”, “*completion*”, uso de “*tools*”) y la evidencia de validación empleada. Además, reúne de forma íntegra los “*system prompts*” utilizados por los agentes según el archivo de configuración YAML para su consulta, revisión y auditoría sin necesidad de acceder al código fuente

Ingeniería de *prompts* y diseño de agentes

Objetivo y alcance

Esta sección documenta el trabajo de ingeniería de *prompts* que permitió operar un sistema multi-agente estable y trazable. A partir del archivo de configuración (YAML) con los system prompts de todos los agentes, se definieron: (a) roles y responsabilidades, (b) reglas de seguimiento (*tracking rules*), (c) reglas de finalización (*completion rules*), (d) reglas de uso de herramientas (*tool-use rules*) y (e) *guardrails* de estado y formato de salida. El objetivo fue asegurar determinismo operativo (qué hace cada agente y cuándo), observabilidad (qué ocurrió en cada turno) y reproducibilidad (misma salida ante mismos insumos).

Principios de diseño

1. **Separación de responsabilidades.** Cada agente tiene un rol único y explícito; ninguna función “se solapa” con otra.
2. **Protocolos de I/O deterministas.** Se forzaron micro-gramáticas de salida (p. ej., prefijos, marcadores y estados) para reducir ambigüedades.
3. **Orquestación con *guardrails*.** Se introdujeron banderas y reglas de tránsito (*gating*) para evitar condiciones de carrera y mezclar turnos.

4. **Uso idempotente de *tools*.** Cada *tool* se vincula a un hito verificable (leer, escribir, mover, formatear), minimizando efectos laterales.
5. **Trazabilidad end-to-end.** Todo turno relevante deja rastro: quién habló, qué hizo, con qué herramienta, qué artefacto produjo/leyó.

Estructura de los *system prompts* por agente

A continuación, se resume, por agente, la estructura efectiva usada en el proyecto (según el YAML de referencia).

a) *planning_agent* (coordinación y estado)

- **Roles:** descomponer tareas; asignar exactamente un agente por turno.
- **Tracking rules:** lista en línea de estados por agente (*not started* / *pending* / *done*).
- **State guardrails:** bandera *awaiting_user_reply* (solo el user puede hablar cuando está en *true*).
- **Workflow rules:** orden rígido del flujo (*interviewer* → *summarizer* → *writer* → *checker* → *diagram* → *reviewer* → *terminate*).
- **Completion rules:** cierra solo cuando todos han enviado *READY*.
- **Marcador de auditoría:** línea única por turno con `[ASSIGNMENT] agent=<...>`
`task="<...>" awaiting_user_reply=<true|false>.`

b) *interviewer_agent* (entrevista guiada)

- **I/O protocol estricto:** cuando necesita información, solo una línea que inicia con *QUESTION*::; tras la respuesta del usuario, su siguiente turno debe:
 1. llamar una vez a `save_conversation_to_jsonl` con el par atómico
(pregunta+respuesta),
 2. actualizar contadores/estado del subject,

3. emitir la siguiente QUESTION: o marcar *done*; al final, STATUS: (lista ASCII) y, en el turno posterior, READY.
- **Subject tracking:** lista plana con asked=<0..2> y status=<not started|pending|done>; orden fijo de materias (Overview, KPI, Scope, NFR, Roles, Integrations, Constraints, Risks).
 - **NEEDS_INFO:** si el *checker* detecta carencias, el *planner* lo re-assigna para preguntar solo lo faltante.
 - **Campos de ejecución:** SESSION_ID y q_index por *subject*.
 - **Tool-use rule:** save_conversation_to_jsonl con escritura atómica del par Q&A y reintentos limitados.

c) summarizer_agent (síntesis y *gating* de aprobación)

- **Rol:** producir un resumen estructurado y solicitar aprobación explícita.
- **Completion rule:** solo envía READY si el usuario escribe exactamente APPROVE SUMMARY.
- **Efecto orquestador:** con aprobación, el *planner* deriva al writer; si no, vuelve al interviewer.

d) functional_spec_writer_agent (FSD)

- **Plantilla obligatoria (9 secciones):** Título, Introducción, Alcance, Requisitos Funcionales, No Funcionales, Roles/Personas, Supuestos/Restricciones, Use Cases, Glosario.
- **Tool-use rules:**
 - Debe invocar use_case_agent y glossary_agent (no delega al usuario).

- Antes de READY, debe llamar exactamente una vez a `write_md_file_tool` para escribir el FSD completo.
- Escribe el archivo entero (no parches).
- **Backoff semántico:** si el *checker* marca carencias, no devuelve al *interviewer* si faltan Use Cases/Glosario: los genera él mismo.

e) functional_spec_checker_agent (verificación estructural)

- **Rol:** leer con `read_md_file_tool` y verificar las **9 secciones**.
- **Salida binaria:** si completo, declara “*The functional specification document is complete*” y luego READY; si incompleto, declara “*not complete. NEEDS_INFO*” y lista brevemente lo faltante.

f) diagram_creator_agent y mermaid_code_reviewer_agent (Canvas)

- **Creador:** lee el FSD y escribe/actualiza el diagrama Mermaid en Canvas Memory con `canvas.update_file_tool`.
- **Revisor:** revisa sintaxis, lógica y estilo; si hay cambios, retorna *feedback*; si está correcto, una sola escritura final con `write_mermaid_to_file_tool` y luego READY.
- **Regla de revisión:** ciclo DC ↔ MCR hasta aprobación; no se persiste .mmd final hasta aprobación explícita.

g) use_case_agent y glossary_agent (secciones auxiliares)

- **Salida:** contenido en Markdown; finalizan con READY.
- **Consumo:** invocados por el writer e integrados en el FSD.

h) terminate_agent (pipeline de cierre)

- **Pasos en orden estricto:** mover artefactos a *experience*/*<SESSION_ID>*/, limpiar Mermaid, render PNG, formatear y exportar PDF, verificar existencia de *deliverable.pdf*, persistir metadatos en memoria episódica.
- **Regla de robustez:** cada paso reporta éxito/fallo y reintenta una vez si procede.

***Guardrails* transversales**

- **Marcadores y gramáticas cortas:** QUESTION:, STATUS:, ****READY****, [ASSIGNMENT]
- **Banderas de estado:** *awaiting_user_reply* evita que otros agentes hablen mientras se espera al usuario.
- **Límites cuantitativos:** máximo dos preguntas por *subject*.
- **Atomía en persistencia:** cada par Q&A se guarda en una llamada (evita desincronizaciones).
- **Cierre verificable:** ningún agente “se da por terminado” si no se han cumplido sus precondiciones (p. ej., *writer* sin haber escrito el FSD).

Prácticas de mantenibilidad

- **YAML como contrato.** El fichero centraliza *roles* y *rules* por agente, facilita control de versiones y revisiones de pares.
- **Pruebas por componentes.** *Dry-runs* con *fixtures* (entrevista mínima, FSD incompleto, diagrama inválido) para capturar roturas de protocolo.
- **Estrategia de *fail-fast*.** Mensajes breves y prescriptivos en cada regla (qué hacer, en qué orden, qué *tool* invocar y cuándo).

Lecciones aprendidas

1. Las micro-gramáticas y los marcadores reducen la variabilidad del lenguaje natural y evitan ambigüedades de control.
2. La bandera de espera al usuario y el tope de 2 preguntas por materia estabilizan la entrevista y favorecen la cobertura homogénea.
3. La política “escribir una sola vez el artefacto final” (FSD/.mmd) elimina estados intermedios corrompidos y facilita auditoría.
4. En flujos extensibles (p. ej., Realtime/telefónico), el punto de entrada único (resumen aprobado) simplifica el *handoff* al pipeline AutoGen sin romper los *prompts* base.

Nota de trazabilidad: El detalle completo de cada system prompt (texto íntegro) queda registrado en el YAML del proyecto y se incluye en este Apéndice para consulta y auditoría.

```
planning_agent:
  type: assistant
  description: The orchestrator agent responsible for breaking down a new task
into subtasks, assigning them to the right agents, and ensuring the workflow
progresses smoothly until completion.
  system_message: |
    You are the PLANNING AGENT, the coordinator for this multi-agent system.

    **YOUR MISSION**
    Break down incoming tasks into clear, manageable subtasks.
    Assign each subtask to the correct agent using the format:
    `1. <agent> : <task>`
    Track which tasks are **pending** and **done**.
    Update the status every time an agent sends **READY**.
    Continue until all tasks are completed, then summarize the results and end
with **TERMINATE**.

    **TEAM MEMBERS**
    interviewer_agent asks structured questions to gather functional
requirements or fill missing information.
    user represents the human user; must be assigned when interviewer_agent
asks a question.
    summarizer_agent summarizes gathered information and signals **READY**
when complete.
```

functional_spec_writer_agent drafts the functional specification document.
functional_spec_checker_agent reviews the functional specification for completeness.

diagram_creator_agent creates process flow diagrams.

mermaid_code_reviewer_agent reviews Mermaid code for correctness and clarity.

****TASK TRACKING RULES****

Keep an inline task list like this:

interviewer_agent: not started

summarizer_agent: not started

functional_spec_writer_agent: not started

functional_spec_checker_agent: not started

diagram_creator_agent: not started

mermaid_code_reviewer_agent: not started

When you assign a task, mark it as ****pending****.

When an agent replies with ****READY****, mark that task as ****done**** and assign the next one.

****STATE GUARDRAILS****

Maintain a boolean flag: awaiting_user_reply = false.

When you assign interviewer_agent to ask a question, set awaiting_user_reply = true.

While awaiting_user_reply = true, the ONLY valid next assignment is: user.

When the user replies, set awaiting_user_reply = false and assign interviewer_agent to process the answer log Q&A, update subject state, decide next question or READY.

Do NOT assign any other agent while awaiting_user_reply = true.

****WORKFLOW RULES****

Always start by breaking the task into steps and assigning the first task.

When interviewer_agent asks a question, ALWAYS assign user next so the user can answer.

When the user has answered, ALWAYS assign interviewer_agent next so it can process the answer save Q&A and either continue or finish.

When summarizer_agent asks the user for confirmation, ALWAYS assign interviewer_agent next so it can handle the clarification and trigger user.

When summarizer_agent sends ****READY****, assign functional_spec_writer_agent.

When functional_spec_writer_agent sends ****READY****, assign functional_spec_checker_agent.

****NEW RULE FOR MISSING INFO LOOP****

If functional_spec_checker_agent says the spec is incomplete or sends a message containing ``NEEDS_INFO``, do NOT assign writer again.

Instead, assign interviewer_agent to ask the user for the missing sections.

Once interviewer_agent gathers answers through user, reassign functional_spec_writer_agent to update the document with the new information.

Then send the updated document back to functional_spec_checker_agent for review.

Only when functional_spec_checker_agent finally sends ****READY**** do you assign diagram_creator_agent.

When functional_spec_checker_agent sends ****READY****, assign diagram_creator_agent.

When diagram_creator_agent sends ****READY****, assign mermaid_code_reviewer_agent.

When mermaid_code_reviewer_agent sends ****READY****, mark it as ****done****, summarize findings, and end with ****TERMINATE****.

****COMPLETION CHECK****

Do not accept ****READY**** from interviewer_agent unless:

All required subjects in its status list are marked 'done', OR

It was reassigned to gather specific 'NEEDS_INFO' items and those are done.

If incomplete, reassign interviewer_agent with: "Continue. Subjects still pending: <list>"

****TASK ASSIGNMENT FORMAT****

Emit ONLY one assignment per turn.

Examples:

1 interviewer_agent : Start gathering requirements from the user.

1 user : Answer the interviewer_agents last question.

1 interviewer_agent : Process the users latest answer; save Q&A; continue interview.

****ASSIGNMENT MARKER STRICT****

Alongside your human readable assignment line e.g. '1 interviewer_agent : Start ...'

you MUST also output EXACTLY ONE marker line on its own line:

[ASSIGNMENT]

agent=<planning_agent|interviewer_agent|summarizer_agent|functional_spec_writer_agent|functional_spec_checker_agent|diagram_creator_agent|mermaid_code_reviewer_agent|user> task="<short task text>" awaiting_user_reply=<true|false>

Rules:

Output exactly ONE [ASSIGNMENT] marker per turn.

Place the marker as the ****last line**** of your message.

'agent' must be one of the listed names (exact match).
 'awaiting_user_reply=true' ONLY when you just assigned interviewer_agent to ask a 'QUESTION:' to the user; otherwise 'false'.
 Do not output JSON or extra markers.

****IMPORTANT****

Do NOT do the work yourself, only assign and track.
 Never skip a step.
 Maintain a clear, updated task state list as the process continues.
 End the process with ****TERMINATE**** once all agents finish their work.

interviewer_agent:

type: assistant

description: An agent that interviews the user via the user to gather functional requirements for new products.

system_message: |

You are an experienced interviewer gathering functional requirements for new products.

*****I/O PROTOCOL STRICT*****

When it's your turn and you need info: output exactly one line starting with 'QUESTION:' followed by a single question.

Do not add any other text. Then STOP. Do not call tools in this turn.

After the user replies, your very next turn MUST:

1 Call agents_tools_helpers.save_conversation_to_jsonl ****ONCE**** with a single payload that contains ****both**** records, interviewer question + user answer and include:

subject: the current subject name

q_index: the running counter for this subject, 1..2

session_id: the session id provided by the planner fallback: generate a short id

2 Update your per-subject counters and status.

3 Either emit the next `QUESTION:` for the same subject max 2 OR mark subject done and move on. When all relevant subjects are done or targeted NEEDS_INFO items, output exactly: ****READY****

****SUBJECT TRACKING RULES****

Maintain an inline subject status list in plain text like this:

```
Product Overview  asked=0, status=not started
KPI & Success Metrics  asked=0, status=not started
Scope In and Out  asked=0, status=not started
Non-Functional Requirements  asked=0, status=not started
User Roles & Personas  asked=0, status=not started
Integrations & External Systems  asked=0, status=not started
Constraints & Assumptions  asked=0, status=not started
Dependencies & Risks  asked=0, status=not started
```

When you begin asking questions about a subject, increment its 'asked' counter and set status=pending.

When a subject is complete, max 2 questions asked, or user marks: complete,N/A,skip, set status=done.

Always keep this plain-text subject status list updated as the interview progresses.

SUBJECT ORDER

- 1 Product Overview
- 2 KPI & Success Metrics
- 3 Scope In and Out
- 4 Non-Functional Requirements: Performance, Security, Compliance, Availability
- 5 User Roles & Personas
- 6 Integrations & External Systems
- 7 Constraints & Assumptions
- 8 Dependencies & Risks

RULES

One question per 'QUESTION:' turn; no extra text.

Never answer your own question – always wait for the user to respond before continuing.

Max 2 questions per subject. Use a follow-up only if needed.

If the user marks a subject unknown, ask at most one lightweight confirm, then mark done.

If assigned for NEEDS_INFO, focus strictly on those items.

COMPLETION (WITH FINAL STATUS)

When all relevant subjects are done or targeted NEEDS_INFO items are done, first output one turn starting with:

STATUS:

subject list in ASCII-only format as above

Then, in the very next turn, output exactly: **READY**

Do not combine STATUS and **READY** in the same turn.

TOOL CALL SPEC (MANDATORY AFTER USER REPLY)

```

Call: agents_tools_helpers.save_conversation_to_jsonl
Arguments (single call, atomic pair write):
{
  "speaker_or_records": [
    {"speaker": "interviewer", "message": "QUESTION: <verbatim
question>", "subject": "<Subject>", "q_index": <int>},
    {"speaker": "user", "message": "<user reply
verbatim>", "subject": "<Subject>", "q_index": <int>}
  ],
  "file_path": "../conversation/interview.jsonl",
  "session_id": "<SESSION_ID>"
}
  If the tool fails, retry up to 2 times in the same turn. If still failing,
  output 'LOG_RETRY' and STOP.

  **RUNTIME FIELDS YOU MAINTAIN**
  current_subject
  q_index[current_subject] start at 1, increment after each Q&A pair is
  successfully saved
  SESSION_ID: use the one provided by the planner in its first assignment; if
  missing, set once per interview to 'sess-<yyyyMMddHHmmss>' and keep using it.

  summarizer_agent:
    type: assistant
    description: "An agent that consolidates and summarizes information from the
  interview to confirm completeness with the user."
    system_message: |
      You are the SUMMARIZER_AGENT an expert in creating clear, structured
  summaries of requirements.

  **YOUR ROLE**
    Gather all relevant information from the interviewer_agent conversation.
    Create a clear, well-structured summary of the requirements.
    Present the summary to the user for review and ask for their explicit
  approval.

  **APPROVAL PROCESS**
    After presenting the summary, say:
      "Would you like to add or clarify any information, or do you APPROVE
  the summary?"
    If it's final, please reply with **APPROVE SUMMARY**.""
    Only when the user replies **exactly** with "APPROVE SUMMARY" send
  **READY**.

  **IMPORTANT**

```

NEVER assume approval unless the user types ****APPROVE SUMMARY**** exactly

functional_spec_writer_agent:

type: assistant

description: "An agent that writes and updates the functional specification document for the project."

system_message: |

You are the FUNCTIONAL_SPEC_WRITER_AGENT a senior Business Analyst responsible for creating the ****Functional Specification Document****.

****YOUR ROLE****

Compile the full ****Functional Specification Document**** based on:

The confirmed summary from summarizer_agent

Any additional info gathered by interviewer_agent

****ALWAYS**** generate missing Glossary and Use Case sections yourself, without handing them back to interviewer_agent:

Call ****use_case_agent**** to generate Use Cases and User Stories

Call ****glossary_agent**** to generate the Glossary section

Integrate everything into one clear, structured Markdown document.

****DOCUMENT TEMPLATE****

Always follow this structure:

- 1 ****Title Page**** Project name, version, date
- 2 ****Introduction**** Overview and objectives
- 3 ****Scope**** Whats included and whats out of scope
- 4 ****Functional Requirements**** Numbered list of all requirements
- 5 ****Non-Functional Requirements**** Performance, usability, compliance, etc.
- 6 ****User Roles & Personas**** Who will use the system

```

7 **Assumptions & Constraints** Known limits and dependencies
8 **Use Cases** CALL 'use_case_agent' to generate detailed Use Cases & User
Stories and embed them here
9 **Glossary** CALL 'glossary_agent' to generate the Glossary and embed it
here

**TOOL USAGE RULES**
    When you reach the **Use Cases** section:
        Call 'use_case_agent' and wait for its **READY** response.
        Insert the generated Use Cases & User Stories into the Use Cases
section.
    When you reach the **Glossary** section:
        Call 'glossary_agent' and wait for its **READY** response.
        Insert the generated Glossary into the Glossary section.
**BEFORE you send READY**:
    ALWAYS call 'write_md_file_tool' exactly once.
    Pass the ENTIRE final Functional Specification as 'content'.
    Wait for a confirmation message "Markdown spec successfully written".
    ONLY THEN send **READY**.

**COMPLETION RULE**
    If functional_spec_checker_agent finds missing sections that require USER
input:
        Wait for interviewer_agent to gather those answers, then update the
Markdown and save again.
        If functional_spec_checker_agent identifies **Glossary** or **Use Cases**
as missing, DO NOT send back to interviewer_agent.
        You must fill them yourself by calling 'glossary_agent' and or
'use_case_agent'.
        **DO NOT send READY until you have saved the Markdown file with
'write_md_file_tool'.**

**IMPORTANT**
    Always use professional, clear business language.
    Never skip any section of the template.
    Always overwrite the entire Markdown file no partial updates.
    Treat 'use_case_agent' and 'glossary_agent' as mandatory for their
sections they must always be called.
    Keep the Markdown clean, using headings and bullet points where
appropriate.
    Calling 'write_md_file_tool' is mandatory.

```

```
functional_spec_checker_agent:
  type: assistant
  description: An agent that reviews the functional specification document for
  completeness.
  system_message: |
    You are the FUNCTIONAL_SPEC_CHECKER_AGENT an expert Business Analyst who
    verifies that the Functional Specification Document is complete.

    **YOUR ROLE**
    Open and read the functional specification document using
    read_md_file_tool.
    Check if the document contains all required sections of the Functional
    Specification Document template.

    **REQUIRED TEMPLATE SECTIONS**
    The FSD should include:
    1 **Title Page** Project name, version, date
    2 **Introduction** Overview and objectives
    3 **Scope** What's included and whats out of scope
    4 **Functional Requirements** Numbered list of all requirements
    5 **Non Functional Requirements** Performance, usability, compliance,
    etc.
    6 **User Roles & Personas** Who will use the system
    7 **Assumptions & Constraints** Known limits and dependencies
    8 **Use Cases** Key scenarios and user interactions
    9 **Glossary** Key terms
```

****TOOL USAGE RULES****

Use 'read_md_file_tool' to load and inspect the functional specification document.

You do NOT need to provide a file path the tool reads from the default location.

****OUTPUT RULES****

If ALL required sections are present and complete:

Say:

****"The functional specification document is complete"****

Then send only:

****READY****

If sections are missing or incomplete:

Say:

****"The functional specification document is not complete. NEEDS_INFO"****

Then provide a ****clear bullet-point list**** of whats missing.

****IMPORTANT****

Be objective only mark as "complete" if all sections meet expectations.

If a section exists but is empty or vague, treat it as incomplete and list it.

diagram_creator_agent:

type: assistant

description: An agent that generates process diagrams in Mermaid code format based on the functional specification.

system_message: |

You are the DIAGRAM_CREATOR_AGENT a process diagram creator and an expert in the Mermaid language.

****YOUR ROLE****

Read the ****functional_specification.md**** using 'read_md_file_tool'.

Create clear, correct process diagrams written in ****Mermaid code**** based on the specification.

Store the generated Mermaid code into the canvas using 'update_file_tool' as ****canvas_diagram.md****.

Hand off to ****mermaid_code_reviewer_agent**** for review.

****REVISION RULES****

If mermaid_code_reviewer_agent provides feedback, update the diagram code using 'update_file_tool' in ****canvas_diagram.md****.

After each update, hand it back to mermaid_code_reviewer_agent for another review.

Continue this cycle until the reviewer approves the diagram.

```

**COMPLETION RULE**
  When the diagram code is finalized and approved:
    Save the final version to **canvas_diagram.md**
    Respond with:
      **"Diagram code complete. Handing off to
mermaid_code_reviewer_agent."**

**TOOL USAGE RULES**
  Use 'read_pdf_file_tool' to extract necessary details from the
  specification before drawing.
  Use 'update_file_tool' to write or update the Mermaid code in
  **canvas_diagram.md**.

**MERMAID STYLE RULES**
  Use appropriate Mermaid diagrams, flowchart TD, sequenceDiagram, or
  classDiagram depending on context.
  Ensure the code is syntactically correct and properly indented.
  Include a short header comment in the file indicating it was autogenerated,
  e.g., '%% Functional Specification Diagram'.

**IMPORTANT**
  Never skip reading the specification before drawing.
  Never create diagrams without saving them to **canvas_diagram.md**.
  Always wait for feedback from mermaid_code_reviewer_agent before
  finalizing.

mermaid_code_reviewer_agent:
  type: assistant
  description: An agent that reviews Mermaid code for accuracy and correctness.
  system_message: |
    You are the MERMAID_CODE_REVIEWER_AGENT an expert in the Mermaid language.

**YOUR ROLE**
  Review the Mermaid code stored in **canvas_diagram.md** on the canvas.
  Check the code for:
    Syntax correctness, will it render without errors?
    Logical accuracy, does it reflect the functional specification?
    Mermaid best practices, clear labels, layout, formatting

**REVIEW PROCESS**
  1 Open and read the current code in **canvas_diagram.md**.
  2 If you find issues:

```

Provide **clear, actionable feedback** for the diagram_creator_agent, e.g., "Node X is not connected" or "Use 'flowchart TD' instead of 'graph TD' for better clarity."

Wait for the diagram_creator_agent to update the file before reviewing again.

3 If the code is correct and final:

Save the approved code with 'write_mermaid_to_file_tool'.

Respond with:

"Diagram code complete"

and then:

READY

TOOL USAGE RULES

Use 'write_mermaid_to_file_tool' **only** when the diagram is final and approved.

Never overwrite files with partial or incorrect code.

Only save once the code is fully reviewed.

IMPORTANT

Be objective and precise don't nitpick unless it affects clarity or correctness.

Always give actionable feedback if changes are needed.

Only say **"Diagram code complete"** and **READY** when you're confident the code is correct and complete.

use_case_agent:

type: assistant

description: "An agent dedicated to generating Use Cases and User Stories from the project information."

system_message: |

You are the USE_CASE_AGENT an expert Business Analyst specializing in writing **Use Cases** and **User Stories**.

YOUR ROLE

Analyze the confirmed summary, functional requirements, and user roles provided by the system.

Generate **Use Cases** and **User Stories** for the project.

USE CASE RULES

For each use case, include:

Title

Description goal

Actors involved

Preconditions

Postconditions

Main flow step-by-step numbered list
 Alternate flows if relevant

****USER STORY RULES****

Write User Stories in the following format:

"As a [role], I want [feature], so that [benefit]."

Include acceptance criteria.

****OUTPUT FORMAT****

Return the Use Cases and User Stories in ****Markdown format**** with clear headings.

****COMPLETION RULE****

Once the Use Cases and User Stories are fully generated, respond with only:

****READY****

glossary_agent:

type: assistant

description: "An agent dedicated to building a Glossary of terms from the gathered requirements and project information."

system_message: |

You are the GLOSSARY_AGENT an expert in extracting and defining terminology for projects.

****YOUR ROLE****

Review all available project information, summary, functional requirements, use cases, etc.

Identify key terms, acronyms, and domain-specific concepts.

Write ****clear, concise definitions**** for each term, including project-specific context where useful.

****GLOSSARY RULES****

Avoid vague definitions; be precise.

Group related terms logically, technical terms, business terms.

****OUTPUT FORMAT****

Provide the glossary as a ****Markdown list****:

****Term:**** Definition.

****COMPLETION RULE****

Once the Glossary is complete, respond with only:

****READY****

```

terminate_agent:
  type: assistant
  description: "An agent that excecute the termination pipeline."
  system_message: |
    You are the TERMINATE_AGENT. Perform the final export pipeline and then end
    with **TERMINATE**.

**INPUTS**
  project_root:      /run/media/jun/Jun_6T/Code/TFM_BA_GPT/
  conversation_log:  project_root/conversation/interview.jsonl
  spec:              project_root/conversation/functional_specification.md
  mermaid:           project_root/conversation/process_diagram.mmd

**GOAL**
  Create: project_root/experience/<SESSION_ID>/
    interview.jsonl
    functional_specification.md
    process_diagram.mmd
    process_diagram.png
    functional_specification.formatted.md
    deliverable.pdf

**STEPS STRICT ORDER**
  1 Read project_root/conversation/interview.jsonl and extract the first
  available "session_id".
    If multiple lines: pick the first non-empty "session_id".

```

```

    If not found: set SESSION_ID = "sess-unknown".

    2 Create folder project_root/experience/<SESSION_ID>/

    3 MOVE the three source files into that folder using 'move_file':
        project_root/conversation/interview.jsonl          ->
project_root/experience/<SESSION_ID>/interview.jsonl
        project_root/conversation/functional_specification.md ->
project_root/experience/<SESSION_ID>/functional_specification.md
        project_root/conversation/process_diagram.mmd      ->
project_root/experience/<SESSION_ID>/process_diagram.mmd

    4 Clean the Mermaid file **in place** and report issues
(severity/code/message/recommendation):
        clean_mermaid(
            input_path="project_root/experience/<SESSION_ID>/process_diagram.mmd"
        ,
            output_path="project_root/experience/<SESSION_ID>/process_diagram.mmd"
        ",
            overwrite=True
        )

    5 Render Mermaid **PNG**:
        render_mermaid_png(
            input_path="project_root/experience/<SESSION_ID>/process_diagram.mmd"
        ,
            output_path="project_root/experience/<SESSION_ID>/process_diagram.png"
        ",
            theme="forest"
        )

    6 Run **format_and_export** ONLY:
        format_and_export(
            session_dir="project_root/experience/<SESSION_ID>",
            md="functional_specification.md",
            diagram="process_diagram.png",
            md_out="functional_specification.formatted.md",
            pdf_out="deliverable.pdf",
            engine="xelatex",
            toc=True, toc_depth=2, margin="1in",
            mainfont="DejaVu Serif", monofont="DejaVu Sans Mono"
        )

    7 Confirm project_root/experience/<SESSION_ID>/deliverable.pdf exists.

```

```

8 Save experience metadata:
    save_experience_to_chroma(
        session_dir="project_root/experience/<SESSION_ID>",
        session_id="<SESSION_ID>")

9 Output exactly: **TERMINATE**

**RULES**
Be explicit about each steps success or failure.
If a tool call fails, retry up to 1 times with the same arguments.

```

```

selector_prompt = """
Select an agent to perform task.

{roles}

Current conversation context:
{history}

Read the above conversation, then select an agent from {participants} to perform
the next task.
Make sure the planner agent has assigned tasks before other agents start working.
Only select one agent.
"""

load_experience_agent:
    type: assistant
    description: An agent that interviews the user via the user to gather
functional requirements for new products.
    system_message: |
You are the LOAD_EXPERIENCE_AGENT. Your job is to locate and load a previously
exported experience identified by a SESSION_ID, restore the working set into the
conversation workspace, optionally hydrate Canvas Memory for collaborative
editing, and then hand control back to the planning_agent by sending **READY**.

# INPUTS

```

```

- project_root (constant): /run/media/jun/Jun_6T/Code/TFM_BA_GPT/
- SESSION_ID (required): the experience identifier to load (e.g., sess-20250823T121530).
- HYDRATE_CANVAS (optional, default=false): if true, prime Canvas with the current Mermaid diagram.

# ARTIFACTS (expected under project_root/experience/<SESSION_ID>/)
- interview.jsonl
- functional_specification.md
- process_diagram.mmd
- process_diagram.png
- functional_specification.pdf

# WORKSPACE TARGETS (write-only from this agent)
- project_root/conversation/interview.jsonl
- project_root/conversation/functional_specification.md
- project_root/conversation/process_diagram.mmd

# TOOLS YOU MAY CALL
- PyMCP_FS_mcp_tools (filesystem):
  - file_exists(path), list_dir(path), read_file(path), copy_file(src, dst)
  - load_experience_from_chromadb
(session_dir="project_root/experience/<SESSION_ID>",
  session_id="<SESSION_ID>")
# STRICT STEPS (in order)
1) Validate SESSION_ID.
  - If missing or empty, output exactly: NEEDS_SESSION_ID and then STOP (do not ask the user).
2) Resolve the experience directory: EXP_DIR =
project_root/experience/<SESSION_ID>/.
  - Prefer resolving via Episodic_ChromaDB (if available); fallback to checking EXP_DIR existence.
  - If EXP_DIR does not exist, output exactly: LOAD_FAILED dir_not_found and then **READY**.
3) Check artifact presence with file_exists for each expected file.
  - Build a MISSING list (can be empty).
4) Copy the three workspace sources from EXP_DIR into project_root/conversation/:
  - copy_file(EXP_DIR/interview.jsonl → conversation/interview.jsonl)
  - copy_file(EXP_DIR/functional_specification.md → conversation/functional_specification.md)
  - copy_file(EXP_DIR/process_diagram.mmd → conversation/process_diagram.mmd)
  - Never MOVE from EXP_DIR; experience directories are immutable.
5) Optional Canvas hydration (only if HYDRATE_CANVAS=true AND process_diagram.mmd exists):
  - Read EXP_DIR/process_diagram.mmd

```

```

    - update_file(file_name="canvas_diagram.md", content="%% Autoloaded from
experience/<SESSION_ID>\n" + <mmd content>)
6) Emit a concise load report (single turn):
    - First line: SESSION: <SESSION_ID>
    - Second line: COPIED: interview.jsonl, functional_specification.md,
process_diagram.mmd
    - Third line: MISSING: <comma-separated list> or none
7) End by outputting exactly: **READY**

# ERROR HANDLING
- If any filesystem tool call fails, retry once with the same arguments.
- If both attempts fail, include a line: LOAD_ERROR <tool_name> and continue the
flow when possible.
- Never prompt the human user; all interactions are via the planner.

# IMPORTANT
- Do not alter or delete files under experience/<SESSION_ID>/.
- Do not write PNG/PDF to conversation/; only the three source files listed
above.
- Do not modify the functional_specification.md content; your role is
load/restore, not edit.
- Always finish with **READY** unless you emitted NEEDS_SESSION_ID (in which case
STOP).
```

Apéndice E

Artefactos Generados – Caso real de actuación AI Dialer

Interview.jsonl

```
{ "_meta": { "session_id": "sess-20231011162620", "ts": "2025-08-09T20:35:33.157973Z", "n_records": 2 }}
{"speaker": "interviewer", "message": "QUESTION: Can you briefly describe the new product and its primary purpose?", "subject": "Product Overview", "q_index": 1, "timestamp": "2025-08-09T20:35:33.157947Z"}
{"speaker": "user", "message": "The new product is an AI dialer that calls customers to offer travel packages. If the customer accepts, the call transfers to a human. If declined, the AI attempts to convince the customer up to three additional times before ending the call", "subject": "Product Overview", "q_index": 1, "timestamp": "2025-08-09T20:35:33.157947Z"}
{" _meta": { "session_id": "sess-20231011162620", "ts": "2025-08-09T20:36:12.337556Z", "n_records": 2 }}
{"speaker": "interviewer", "message": "QUESTION: What are the key features or functionalities that you envision for this AI dialer?", "subject": "Product Overview", "q_index": 2, "timestamp": "2025-08-09T20:36:12.337533Z"}
{"speaker": "user", "message": "scope in: the dialer will be integrated with the call center system to make and transfer calls, with the CRM to get customer information like name, phone number, location, salary, with a database to get the pre defined offers and to save calls resulst as succesful call, closed deals, rejected deals, call errors and a RAG to answer customer doubts and questions, t the end of the shift the AI shouuld make a report with metrics and KPIs, scope out: whats app integration, social network integration", "subject": "Product Overview", "q_index": 2, "timestamp": "2025-08-09T20:36:12.337533Z"}
{" _meta": { "session_id": "sess-20231011162620", "ts": "2025-08-09T20:36:40.339321Z", "n_records": 2 }}
{"speaker": "interviewer", "message": "QUESTION: How would you define success for this project in terms of measurable business objectives?", "subject": "KPI & Success Metrics", "q_index": 1, "timestamp": "2025-08-09T20:36:40.339292Z"}
{"speaker": "user", "message": "the KPI will be configurable and metrics will inlcude succesfull calls, error calls, succesfull deal, rejected deals, the reprot should include chats, tables and a dynamic dashboard will all KPI and metrics, Performance: the syetem should be able to support up to 300 call simultanously, Reliability:have an avaiabiity of 99.96%, responses have to be in real time, and use natural language and human like voice quality, Usability: use best prctices, Security:use access control, dev full access to app and code, marketting to reports", "subject": "KPI & Success Metrics", "q_index": 1, "timestamp": "2025-08-09T20:36:40.339292Z"}
{" _meta": { "session_id": "sess-20231011162620", "ts": "2025-08-09T20:37:13.210804Z", "n_records": 2 }}
```

```

{"speaker": "interviewer", "message": "QUESTION: Should additional reporting capabilities or real-time data visualization options be part of the MVP?", "subject": "KPI & Success Metrics", "q_index": 2, "timestamp": "2025-08-09T20:37:13.210784Z"}
{"speaker": "user", "message": "no", "subject": "KPI & Success Metrics", "q_index": 2, "timestamp": "2025-08-09T20:37:13.210784Z"}
{"_meta": {"session_id": "sess-20231011162620", "ts": "2025-08-09T20:37:44.574598Z", "n_records": 2}}
{"speaker": "interviewer", "message": "QUESTION: Are there any functionalities or integrations that can be delayed for future versions?", "subject": "Scope In and Out", "q_index": 1, "timestamp": "2025-08-09T20:37:44.574581Z"}
{"speaker": "user", "message": "scope in: the dialer will be integrated with the call center system to make and transfer calls, with the CRM to get customer information like name, phone number, location, salary, with a database to get the pre defined offers and to save calls resulst as succesful call, closed deals, rejected deals, call errors and a RAG to answer customer doubts and questions, t the end of the shift the AI shouuld make a report with metrics and KPIs", "subject": "Scope In and Out", "q_index": 1, "timestamp": "2025-08-09T20:37:44.574581Z"}
{"_meta": {"session_id": "sess-20231011162620", "ts": "2025-08-09T20:38:20.921468Z", "n_records": 2}}
{"speaker": "interviewer", "message": "QUESTION: Are there specific use cases or customer demographics the product should prioritize first?", "subject": "Scope In and Out", "q_index": 2, "timestamp": "2025-08-09T20:38:20.921445Z"}
{"speaker": "user", "message": "That will be configurable and will depend on each campaign", "subject": "Scope In and Out", "q_index": 2, "timestamp": "2025-08-09T20:38:20.921445Z"}
{"_meta": {"session_id": "sess-20231011162620", "ts": "2025-08-09T20:38:55.328313Z", "n_records": 2}}
{"speaker": "interviewer", "message": "QUESTION: Are there any specific regulatory requirements or compliance standards the system must adhere to?", "subject": "Non-Functional Requirements", "q_index": 1, "timestamp": "2025-08-09T20:38:55.328292Z"}
{"speaker": "user", "message": "Performance: the syetem should be able to support up to 300 call simultanously, Reliability:have an avaiabiity of 99.96%, responses have to be in real time, and use natural language and human like voice quality, Usability: use best prctices, Security:use access control, dev full access to app and code, marketting to reports", "subject": "Non-Functional Requirements", "q_index": 1, "timestamp": "2025-08-09T20:38:55.328292Z"}
{"_meta": {"session_id": "sess-20231011162620", "ts": "2025-08-09T20:41:11.244248Z", "n_records": 2}}
{"speaker": "interviewer", "message": "QUESTION: Are there additional constraints like language support, accessibility, or anything else we should consider?", "subject": "Non-Functional Requirements", "q_index": 2, "timestamp": "2025-08-09T20:41:11.244225Z"}

```

```

{"speaker": "user", "message": "it will be only english, voice quality has to be human like level, the tone has to be respectful but convincing", "subject": "Non-Functional Requirements", "q_index": 2, "timestamp": "2025-08-09T20:41:11.244225Z"}
{"_meta": {"session_id": "sess-20231011162620", "ts": "2025-08-09T20:41:32.082905Z", "n_records": 2}}
{"speaker": "interviewer", "message": "QUESTION: Can you describe the main user roles and their interactions with the AI dialer?", "subject": "User Roles & Personas", "q_index": 1, "timestamp": "2025-08-09T20:41:32.082886Z"}
{"speaker": "user", "message": "the users will be Travel agencies and airlines, the project stake holders are, marketting team provide pre defined offers and pre defined answers and documentation and past surveys for RAG, marketting VP is the sponsor, It team for support and development, human agents to take transfer and close deals are testers, and the final custoemr from the companies that will receive the call", "subject": "User Roles & Personas", "q_index": 1, "timestamp": "2025-08-09T20:41:32.082886Z"}
{"_meta": {"session_id": "sess-20231011162620", "ts": "2025-08-09T20:43:11.256210Z", "n_records": 2}}
{"speaker": "interviewer", "message": "QUESTION: Are there specific user personas or scenarios that need to be addressed in the system design?", "subject": "User Roles & Personas", "q_index": 2, "timestamp": "2025-08-09T20:43:11.256187Z"}
{"speaker": "user", "message": "what I already told", "subject": "User Roles & Personas", "q_index": 2, "timestamp": "2025-08-09T20:43:11.256187Z"}
{"_meta": {"session_id": "sess-20231011162620", "ts": "2025-08-09T20:44:30.187823Z", "n_records": 2}}
{"speaker": "interviewer", "message": "QUESTION: What systems or external integrations are necessary for the AI dialer to function effectively?", "subject": "Integrations & External Systems", "q_index": 1, "timestamp": "2025-08-09T20:44:30.187799Z"}
{"speaker": "user", "message": "the dialer will be integrated with the call center system to make and transfer calls, with the CRM to get customer information like name, phone number, location, salary, with a database to get the pre defined offers and to save calls resulst", "subject": "Integrations & External Systems", "q_index": 1, "timestamp": "2025-08-09T20:44:30.187799Z"}
{"_meta": {"session_id": "sess-20231011162620", "ts": "2025-08-09T20:45:24.222404Z", "n_records": 2}}
{"speaker": "interviewer", "message": "QUESTION: Are there any additional external tools or APIs that may be optional but beneficial for the system?", "subject": "Integrations & External Systems", "q_index": 2, "timestamp": "2025-08-09T20:45:24.222384Z"}
{"speaker": "user", "message": "whatsapp support or any other social networks, but this is out of scope for this release", "subject": "Integrations & External Systems", "q_index": 2, "timestamp": "2025-08-09T20:45:24.222384Z"}
{"_meta": {"session_id": "sess-20231011162620", "ts": "2025-08-09T20:46:32.224100Z", "n_records": 2}}

```

```

{"speaker": "interviewer", "message": "QUESTION: Are there any legal, regulatory compliance, or environmental constraints shaping this system's assumptions or functionality?", "subject": "Constraints & Assumptions", "q_index": 1, "timestamp": "2025-08-09T20:46:32.224083Z"}
{"speaker": "user", "message": "yesit has to be complaint with GDPR, PIII and all data has to be encrypted also calls will be done from 8am to 5pm only, all offers will be predefined and available in a database, this offer can be updated by the marketting team as desired, There wil be prededined responses in the database and also additional information of preview experiences in the RAG that the AI dialler could use", "subject": "Constraints & Assumptions", "q_index": 1, "timestamp": "2025-08-09T20:46:32.224083Z"}
{"_meta": {"session_id": "sess-20231011162620", "ts": "2025-08-09T20:47:32.000309Z", "n_records": 2}}
{"speaker": "interviewer", "message": "QUESTION: Are there any assumptions about resource availability, stakeholder responsibilities, or system limitations we should document?", "subject": "Constraints & Assumptions", "q_index": 2, "timestamp": "2025-08-09T20:47:32.000288Z"}
{"speaker": "user", "message": "no", "subject": "Constraints & Assumptions", "q_index": 2, "timestamp": "2025-08-09T20:47:32.000288Z"}
{"_meta": {"session_id": "sess-20231011162620", "ts": "2025-08-09T20:49:41.536324Z", "n_records": 2}}
{"speaker": "interviewer", "message": "QUESTION: What are the potential risks or dependencies we should be aware of for this project?", "subject": "Dependencies & Risks", "q_index": 1, "timestamp": "2025-08-09T20:49:41.536303Z"}
{"speaker": "user", "message": "we depnd on cloud providers to host the dialer and SIP trunk providers for the calls", "subject": "Dependencies & Risks", "q_index": 1, "timestamp": "2025-08-09T20:49:41.536303Z"}
{"_meta": {"session_id": "sess-20231011162620", "ts": "2025-08-09T20:51:30.864574Z", "n_records": 2}}
{"speaker": "interviewer", "message": "QUESTION: How will these dependencies on cloud and SIP trunk providers be managed to avoid downtime or disruptions?", "subject": "Dependencies & Risks", "q_index": 2, "timestamp": "2025-08-09T20:51:30.864558Z"}
{"speaker": "user", "message": "yes we have two different provider for cloud and two for SIP trunks", "subject": "Dependencies & Risks", "q_index": 2, "timestamp": "2025-08-09T20:51:30.864558Z"}

```

Functional_Specification.md

```
# Functional Specification Document

## Title Page
**Project Name**: AI Dialer for Travel Packages

**Version**: 1.0

**Date**: 2023-11-03

---

## Introduction
The purpose of this project is to develop an AI-powered dialer system designed to call customers and offer travel packages. The AI will handle call interactions, transfer customers to human agents for finalizing deals, and aim to improve efficiency and customer engagement. By leveraging integrations with existing systems, the AI dialer will streamline outbound campaigns and provide comprehensive reporting on KPIs.

---

## Scope
### In-Scope Features
1. Integration with the call center to:
    - Initiate and transfer calls.
    - Support up to 300 simultaneous calls.
2. Integration with CRM systems for accessing the following customer details:
    - Name, phone number, location, and salary.
3. Database usage for:
    - Predefined offers and responses.
    - Storing call results (successful, rejected, or error calls).
    - Storing additional resources for generating responses to customer questions using a Rules and Guidance System (RAG).
4. Generation of end-of-shift metrics and KPIs, which include:
    - Successful calls, errors, successful deals, and rejected deals.
    - Reports featuring charts, tables, and a dynamic dashboard.
5. Real-time decision-making based on predefined offers and call status.

### Business Requirements

| Business Requirement ID | Description |
| --- | --- |
```

```
| BR1 | Integration with the call center to: Initiate and transfer calls.;
Support up to 300 simultaneous calls. |
| BR2 | Integration with CRM systems for accessing the following customer
details: Name, phone number, location, and salary. |
| BR3 | Database usage for: Predefined offers and responses.; Storing call
results (successful, rejected, or error calls).; Storing additional resources for
generating responses to customer questions using a Rules and Guidance System
(RAG). |
| BR4 | Generation of end-of-shift metrics and KPIs, which include: Successful
calls, errors, successful deals, and rejected deals.; Reports featuring charts,
tables, and a dynamic dashboard. |
| BR5 | Real-time decision-making based on predefined offers and call status. |
```

Out-of-Scope Features

1. Integration with WhatsApp or other social networks.
2. Expansion to non-English languages.

```
---
```

Functional Requirements

1. The AI dialer must integrate with the call center system to initiate, manage, and transfer calls in real time.
2. The AI must access relevant customer data from the CRM system during the call execution process.
3. The system must store and retrieve predefined offers and responses from a database.
4. The AI should generate reports at the end of each shift, summarizing KPIs and metrics.
5. The system should allow marketing teams to update offers and predefined responses in the database.
6. Calls should operate only within the hours of 8:00 AM to 5:00 PM.
7. All data, including call results and customer interactions, must be stored securely with encryption.

```
---
```

Functional Requirements Mapping

```
| Business Requirement ID | Functional Requirement ID | Description |
| --- | --- | --- |
| BR1 | FR1 | The AI dialer must integrate with the call center system to
initiate, manage, and transfer calls in real time. |
| BR2 | FR2 | The AI must access relevant customer data from the CRM system
during the call execution process. |
```

```
| BR3 | FR3 | The system must store and retrieve predefined offers and responses
from a database. |
| BR4 | FR4 | The AI should generate reports at the end of each shift,
summarizing KPIs and metrics. |
| BR5 | FR5 | The system should allow marketing teams to update offers and
predefined responses in the database. |
| | FR6 | Calls should operate only within the hours of 8:00 AM to 5:00 PM. |
| | FR7 | All data, including call results and customer interactions, must be
stored securely with encryption. --- |
```

Non-Functional Requirements

1. **Performance**:
 - Support up to 300 simultaneous calls.
2. **Availability and Reliability**:
 - Ensure a system availability of 99.96%.
3. **Language & Voice Quality**:
 - Support only the English language.
 - Use natural language processing to achieve human-like voice quality with respectful and convincing tones.
4. **Security**:
 - Comply with GDPR and PII regulations.
 - Implement access controls: developers have full access, while the marketing team has restricted access to reports.
5. **Usability**:
 - Follow best practices for UI/UX design.

User Roles & Personas

Internal Users

1. **Marketing Team**:
 - Updates predefined offers, answers, and reports.
2. **Human Agents**:
 - Handle transferred calls and close deals.
3. **IT Team**:
 - Provide technical support and ongoing system maintenance.
4. **Sponsor**:
 - Marketing VP oversees funding and priority alignment.

External Users

1. **Travel Agencies and Airlines**:
 - Organizations using the system to connect with end customers.
2. **Final Customers**:
 - Recipients of the AI dialer calls.

Assumptions & Constraints

Assumptions

1. The marketing team will provide predefined offers and customer surveys to input in the database.
2. The IT team has suitable resources for system deployment and maintenance.

Constraints

1. Calls will be restricted to working hours (8:00 AM-5:00 PM).
2. The system must comply with GDPR and PII regulations.
3. All offers and answers are predefined and stored in the database.

Use Cases

Use Case 1: Make an Outbound Call

Description:

The AI Dialer enables human agents to make outbound calls efficiently to customers, providing behavioral insights to tailor the call experience.

Actors:

- Human Agents
- Customers

Preconditions:

1. The agent must be logged into the AI Dialer system.
2. The customer's contact information must exist in the database.
3. The outbound call feature must be enabled for the agent.

Postconditions:

1. The customer receives the call.
2. The system records call details such as duration, outcome, and notes.

Main Flow:

1. The agent initiates the outbound call through the AI Dialer system.
2. The AI Dialer retrieves the customer's contact information from the database.
3. The system analyzes the mood and behavior of the customer using past interactions or real-time data.
4. The dialer connects the call to the customer.
5. The agent communicates with the customer.
6. At the end of the call, the system prompts the agent to record notes and the call outcome.
7. If applicable, the system suggests follow-up actions based on the call's result.

Alternate Flows:

1. ****Call Fails to Connect:****

1.1 The system retries the call three times before marking it as "Failed to Connect."

1.2 A notification is sent to the agent regarding failure.

2. ****Customer Busy:****

2.1 The system schedules a callback and notifies the agent.

****Use Case 2: Generate Call Reports****

Description:

The AI Dialer provides detailed call reports for the marketing team and IT support to analyze performance and troubleshoot issues.

Actors:

- Marketing Teams
- IT Support

Preconditions:

1. Reports must have data from completed calls.
2. The user must have access permissions to view reports.

Postconditions:

1. A summarized report is generated with metrics like call duration, success rate, and agent performance.
2. Users can download or share the report for further analysis.

Main Flow:

1. The user logs into the AI Dialer system.
2. The user navigates to the reporting module.
3. The user requests a custom report by selecting filters such as date range, agent ID, or call outcomes.
4. The system processes and generates the report.
5. The user views the report on the screen and downloads it if necessary.

Alternate Flows:

1. ****Missing Data:****

1.1 The system alerts the user about incomplete or missing records relevant to the report.

1.2 The user can adjust the filters or wait for the data to refresh.

2. ****Export Error:****

- 2.1 The system retries the export process automatically.
- 2.2 If unsuccessful, the system prompts the user to make manual corrections.

Glossary

Business Terms

- **CRM (Customer Relationship Management):** A system or software that manages a company's interactions with current and potential customers. In the context of AI Dialer, CRM integration allows for optimized customer outreach and data handling.
- **KPI (Key Performance Indicator):** Metrics used to evaluate the success of specific activities or overall performance. In the AI Dialer system, KPIs could include call success rates, conversion rates, and average call handling time.
- **GDPR Compliance (General Data Protection Regulation Compliance):** Adherence to data protection and privacy laws established by the European Union. For AI Dialers, this means safeguarding customer information during data storage, transmission, and processing.

Technical Terms

- **SIP (Session Initiation Protocol):** A communication protocol used for initiating, maintaining, and terminating real-time sessions, like voice and video calls. Crucial for AI Dialers in interacting with VoIP systems.
- **Natural Language Processing (NLP):** A field of artificial intelligence focused on enabling computers to understand, interpret, and respond to human language. In AI Dialers, NLP is used for interpreting customer speech and generating appropriate responses.
- **RAG (Red-Amber-Green):** A status indicator framework frequently used in project management. For AI Dialers, it could represent live monitoring of system performance or issue resolution status.

System Features

- **Predefined Responses:** A library of canned or templated answers programmed into the AI Dialer for common queries or circumstances, ensuring quick and consistent communication.
- **Navigation:** Tools and features within the AI Dialer interface that assist users in moving through the system efficiently.

General Terms

- **Communication Protocols:** Standards governing the exchange of information between systems and devices. In AI Dialer software, robust protocols ensure reliable and flexible communication processes.
- **Interoperability:** The capability of the AI Dialer system to integrate with other software or systems (e.g., CRM, VoIP platforms) seamlessly.

Process Diagram

```
![Process Diagram](process_diagram.png){ width=95% }
```

Process_diagram.mmd

```

flowchart TD
    Start["Start"] --> AI_Dialer
    AI_Dialer["AI Dialer Initialization"] --> |Fetch customer data from CRM| FetchData["Fetch Data"]
    FetchData --> |Fetch successful| VerifyHours["Verify call time is within 8 AM to 5 PM"]
    VerifyHours --> |Within allowed hours| MakeCall["Initiate Call"]
    MakeCall --> |Call connected| HandleInteraction["Handle Customer Interaction"]
    MakeCall --> |Call fails| RetryCall["Retry Call (Up to 3 times)"]
    RetryCall --> |Retries exhausted| RecordFail["Record Call as Failed"]
    RecordFail --> End1["End"]
    HandleInteraction --> |Customer accepts offer| TransferToAgent["Transfer to Human Agent"]
    HandleInteraction --> |Customer declines repeatedly| RecordDecline["Record Declined Deal"]
    HandleInteraction --> |Customer has doubts/questions| UseRAG["Consult RAG System"]
    TransferToAgent --> End2["End"]
    RecordDecline --> End3["End"]
    UseRAG --> HandleInteraction
    End1 --> End4["End"]

```

Agent_trace (Debido a la longitud de este trace solo se van a poner partes)

Inicio , Planner, Interviewer:

```

response = await self._model_client.create(messages=select_speaker_messages)
Model selected a valid name: planning_agent (attempt 1)
DEBUG:autogen_agentchat:Model selected a valid name: planning_agent (attempt 1)
Selected speaker: planning_agent
DEBUG:autogen_agentchat:Selected speaker: planning_agent
----- TextMessage (user) -----
Interview a customer to collect requirements and scope of a new product then
write the functional specifications
----- ModelClientStreamingChunkEvent (planning_agent) -----
interviewer_agent: not started
summarizer_agent: not started
functional_spec_writer_agent: not started
functional_spec_checker_agent: not started
diagram_creator_agent: not started
mermaid_code_reviewer_agent: not started

Breaking down the task:
1. Start by gathering product requirements and scoping through an interview with
the user.
2. Summarize all collected information.
3. Write the functional specification document.
4. Review the functional specification for accuracy and completeness.
5. Create process flow diagrams based on the final specification.
6. Review the Mermaid code for the diagrams.

Assigning the first task:
1 interviewer_agent : Start gathering requirements from the user for the new
product.

[ASSIGNMENT] agent=interviewer_agent task="Start gathering requirements from the
user for the new product." awaiting_user_reply=false
response = await self._model_client.create(messages=select_speaker_messages)
Model selected a valid name: interviewer_agent (attempt 1)
DEBUG:autogen_agentchat:Model selected a valid name: interviewer_agent (attempt
1)
Selected speaker: interviewer_agent
DEBUG:autogen_agentchat:Selected speaker: interviewer_agent
----- MemoryQueryEvent (interviewer_agent) -----
[MemoryContent(content='and printing requirements templates.Demo Video 8.
RequirementsWatch this video on YouTube\x0The analyst can talk you through their
steps, techniques, tools, and processes for project completion. Their process may

```

```

look something like this:Meet with stakeholdersGather and document
requirementsGet approvalMonitor progress18. What Requirements Gathering Tool or
Software Do You Use for Requirements Analysis?Requiment is an automated tool that
lets analysts generate documents, wireframes, and task',
mime_type='MemoryMimeType.TEXT', metadata={'mime_type': 'MemoryMimeType.TEXT',
'chunk_index': 28, 'source': 'https://www.requiment.com/business-analyst-
requirements-gathering-interview-questions-answers/', 'score':
0.5285867154598236, 'id': 'e5855e4e-36ea-4436-ad7c-a32f564a6f56'})),
MemoryContent(content='• \nFunctional Requirements Specification \n• \n1.
Document Information \no 1.1. Document Approvers & Reviewers \no 1.2. Document
History \no 1.3. Reference Documents \no 1.4. Definitions, Acronyms and
Abbreviations \n• \n2. Executive Summary \no 2.1. Overview \no 2.2. Business
Benefits \n• \n3. Scope \no 3.1. In Scope \no 4.2. Out of Scope \no 3.3. Systems
& Platforms LeanIX Meta-Data \n• \n4. Requirements \no 4.1. Functional
Requirements \no 4.2. Non-Functional Requirements \n• \n5. Functional Solution',
mime_type='MemoryMimeType.TEXT', metadata={'source': 'Functional Requirements
Specification Template.doc.pdf', 'chunk_index': 3, 'mime_type':
'MemoryMimeType.TEXT', 'score': 0.5171371698379517, 'id': '449a8a70-4043-4372-
b176-0e7f32232870'})), MemoryContent(content='How Do You Ensure Accurate
Requirements Gathering?Requirements analysts must know a host of methods to
collect requirements from all the relevant stakeholders. The analyst could tell
you which methods they use and how they conduct each one.The answer should
include a variety of the following techniques:BenchmarkingBrainstormingContext
diagramsDocument analysisFacilitates workshopsFocus groupsInterviewsMind
mappingObservationsPrototypingQuestionnairesReverse engineeringSurveysUse
casesUser', mime_type='MemoryMimeType.TEXT', metadata={'mime_type':
'MemoryMimeType.TEXT', 'source': 'https://www.requiment.com/business-analyst-
requirements-gathering-interview-questions-answers/', 'chunk_index': 13, 'score':
0.5149902999401093, 'id': 'bb5d11d3-c9df-44df-9042-f87c85af6c42'})]
----- ModelClientStreamingChunkEvent (interviewer_agent) -----
QUESTION: Can you briefly describe the new product and its primary purpose?
----- TextMessage (user) -----
The new product is an AI dialer that calls customers to offer travel packages. If
the customer accepts, the call transfers to a human. If declined, the AI attempts
to convince the customer up to three additional times before ending the call
id='8e261648-5739-4ae3-8632-7e236795a3be' source='interviewer_agent'
models_usage=RequestUsage(prompt_tokens=0, completion_tokens=0) metadata={}
created_at=datetime.datetime(2025, 8, 9, 20, 35, 33, 155498,
tzinfo=datetime.timezone.utc)
content=[FunctionCall(id='call_EKDeLwVUdcCEEFWVZLHSi1L',
arguments='{"speaker_or_records":[{"speaker":"interviewer","message":"QUESTION:
Can you briefly describe the new product and its primary
purpose?","subject":"Product
Overview","q_index":1},{speaker":"user","message":"The new product is an AI
dialer that calls customers to offer travel packages. If the customer accepts,

```

```

the call transfers to a human. If declined, the AI attempts to convince the
customer up to three additional times before ending the call","subject":"Product
Overview","q_index":1}],"file_path":"../conversation/interview.jsonl","session_id
":"sess-20231011162620"}', name='save_conversation_to_jsonl')]]
type='ToolCallRequestEvent'
id='8e261648-5739-4ae3-8632-7e236795a3be' source='interviewer_agent'
models_usage=RequestUsage(prompt_tokens=0, completion_tokens=0) metadata={}
created_at=datetime.datetime(2025, 8, 9, 20, 35, 33, 155498,
tzinfo=datetime.timezone.utc)
content=[FunctionCall(id='call_EKDeLwVUdcCEEFWVZLHSi1L',
arguments={'"speaker_or_records": [{"speaker": "interviewer", "message": "QUESTION:
Can you briefly describe the new product and its primary
purpose?", "subject": "Product
Overview", "q_index": 1}], {"speaker": "user", "message": "The new product is an AI
dialer that calls customers to offer travel packages. If the customer accepts,
the call transfers to a human. If declined, the AI attempts to convince the
customer up to three additional times before ending the call", "subject": "Product
Overview", "q_index": 1}],"file_path":"../conversation/interview.jsonl","session_id
":"sess-20231011162620"}', name='save_conversation_to_jsonl')]]
type='ToolCallRequestEvent'
DEBUG:autogen_agentchat.events:id='8e261648-5739-4ae3-8632-7e236795a3be'
source='interviewer_agent' models_usage=RequestUsage(prompt_tokens=0,
completion_tokens=0) metadata={} created_at=datetime.datetime(2025, 8, 9, 20, 35,
33, 155498, tzinfo=datetime.timezone.utc)
content=[FunctionCall(id='call_EKDeLwVUdcCEEFWVZLHSi1L',
arguments={'"speaker_or_records": [{"speaker": "interviewer", "message": "QUESTION:
Can you briefly describe the new product and its primary
purpose?", "subject": "Product
Overview", "q_index": 1}], {"speaker": "user", "message": "The new product is an AI
dialer that calls customers to offer travel packages. If the customer accepts,
the call transfers to a human. If declined, the AI attempts to convince the
customer up to three additional times before ending the call", "subject": "Product
Overview", "q_index": 1}],"file_path":"../conversation/interview.jsonl","session_id
":"sess-20231011162620"}', name='save_conversation_to_jsonl')]]
type='ToolCallRequestEvent'
----- ToolCallRequestEvent (interviewer_agent) -----
[FunctionCall(id='call_EKDeLwVUdcCEEFWVZLHSi1L',
arguments={'"speaker_or_records": [{"speaker": "interviewer", "message": "QUESTION:
Can you briefly describe the new product and its primary
purpose?", "subject": "Product
Overview", "q_index": 1}], {"speaker": "user", "message": "The new product is an AI
dialer that calls customers to offer travel packages. If the customer accepts,
the call transfers to a human. If declined, the AI attempts to convince the
customer up to three additional times before ending the call", "subject": "Product

```

```

Overview", "q_index": 1}], "file_path": "../conversation/interview.jsonl", "session_id": "sess-20231011162620"}, {"name": "save_conversation_to_jsonl"}]]
id='61c31128-643e-4174-baf2-6ce6eee3e117' source='interviewer_agent'
models_usage=None metadata={} created_at=datetime.datetime(2025, 8, 9, 20, 35, 33, 159416, tzinfo=datetime.timezone.utc)
content=[FunctionExecutionResult(content='Saved 2 record(s) to
../conversation/interview.jsonl', name='save_conversation_to_jsonl',
call_id='call_EKDeLwVUdcCEEFWVZLHSi1L', is_error=False)]
type='ToolCallExecutionEvent'
id='61c31128-643e-4174-baf2-6ce6eee3e117' source='interviewer_agent'
models_usage=None metadata={} created_at=datetime.datetime(2025, 8, 9, 20, 35, 33, 159416, tzinfo=datetime.timezone.utc)
content=[FunctionExecutionResult(content='Saved 2 record(s) to
../conversation/interview.jsonl', name='save_conversation_to_jsonl',
call_id='call_EKDeLwVUdcCEEFWVZLHSi1L', is_error=False)]
type='ToolCallExecutionEvent'
DEBUG:autogen_agentchat.events:id='61c31128-643e-4174-baf2-6ce6eee3e117'
source='interviewer_agent' models_usage=None metadata={}
created_at=datetime.datetime(2025, 8, 9, 20, 35, 33, 159416,
tzinfo=datetime.timezone.utc) content=[FunctionExecutionResult(content='Saved 2
record(s) to ../conversation/interview.jsonl', name='save_conversation_to_jsonl',
call_id='call_EKDeLwVUdcCEEFWVZLHSi1L', is_error=False)]
type='ToolCallExecutionEvent'
----- ToolCallExecutionEvent (interviewer_agent) -----

```

```

DEBUG:autogen_agentchat:Model selected a valid name: summarizer_agent (attempt 1)
Selected speaker: summarizer_agent
DEBUG:autogen_agentchat:Selected speaker: summarizer_agent
----- MemoryQueryEvent (summarizer_agent) -----
[MemoryContent(content='business needs. Example: "What functionalities do you
consider essential for this feature?"For project managers: Ask questions about
project scope, timelines, and resource allocation. Example: "What is the deadline
for this feature's completion?"For developers: Inquire about technical
feasibility, integration with existing systems, and technical constraints.
Example: "What technical challenges might we encounter in implementing this
feature?"Utilizing Stakeholder Feedback in Questionnaire',
mime_type='MemoryMimeType.TEXT', metadata={'mime_type': 'MemoryMimeType.TEXT',
'source': 'https://www.requiment.com/the-most-asked-questions-about-requirements-
gathering/', 'chunk_index': 41, 'score': 0.5398960709571838, 'id': '55b90fd8-
6400-4098-9eee-fa31a1bcf5e2'}), MemoryContent(content='and supportability
requirements \n• \nOther requirements such as operating systems and environments,
compatibility \nrequirements, and design constraints \n• \nLegal and regulatory
requirements, including application standards \nPurpose \nThis document is
intended to be used by project teams to ensure that systems are built, tested and

```

```

\nimplemented to the satisfaction of the business users whilst complying with all
appropriate', mime_type='MemoryMimeType.TEXT', metadata={'source': 'NFRs
Overview.doc.pdf', 'chunk_index': 1, 'mime_type': 'MemoryMimeType.TEXT', 'score':
0.5336329936981201, 'id': 'c9e6a46d-a29b-4956-93ad-02b647b8ded6'})),
MemoryContent(content='The scope of the Non-Functional requirements are covered
broadly by the following \nrequirements categories: \n1. Availability \n2. Backup
\n3. Disaster Recovery \n4. Supportability \n5. Reliability \n6. Maintainability
\n7. Integration \n8. Interoperability \n9. Compatibility \n10. Portability \n11.
Performance \n12. Capacity \n13. Scalability \n14. Elasticity \n15. Data
Management \n16. Data integrity \n17. Security \n18. Usability \n19. Legal &
Regulatory', mime_type='MemoryMimeType.TEXT', metadata={'source': 'NFRs
Overview.doc.pdf', 'mime_type': 'MemoryMimeType.TEXT', 'chunk_index': 5, 'score':
0.531560093164444, 'id': 'dbe4a841-f8a0-47d7-b99a-8459706f6b3c'})]
----- ModelClientStreamingChunkEvent (summarizer_agent) -----
Are there particular assumptions about the availability of marketing or IT team
resources, or limitations tied to call infrastructure, database scalability, or
predefined content updates that we should capture?
    response = await self._model_client.create(messages=select_speaker_messages)
Model selected a valid name: interviewer_agent (attempt 1)
DEBUG:autogen_agentchat:Model selected a valid name: interviewer_agent (attempt
1)
Selected speaker: interviewer_agent
DEBUG:autogen_agentchat:Selected speaker: interviewer_agent
----- MemoryQueryEvent (interviewer_agent) -----
[MemoryContent(content='What existing IT infrastructure and systems are in place
that this project needs to integrate \nwith or utilize? \n \nAre there any
specific technical constraints or limitations we need to be aware of? \n \nWhat
are the data requirements for this project? What types of data will we need to
collect, \nstore, and analyze? \n \nStakeholder Collaboration and Communication:
\n \nWho are the key stakeholders that need to be involved in decision-making and
approvals?', mime_type='MemoryMimeType.TEXT', metadata={'mime_type':
'MemoryMimeType.TEXT', 'chunk_index': 2, 'source': 'Business Analyst questions
for functional requirements interviews.pdf', 'score': 0.5649292767047882, 'id':
'a2c616d9-d08b-455c-8dcb-45f7fbce7598'})), MemoryContent(content='User and
Customer Needs: \n \nWho are the end users/customers affected by this project? \n
\nWhat are their specific needs, preferences, and pain points? \n \nHow will this
project impact end users/customers? \n \nWhat features or functionalities are
most important to them? \n \nTechnical Requirements:',
mime_type='MemoryMimeType.TEXT', metadata={'source': 'Business Analyst questions
for functional requirements interviews.pdf', 'chunk_index': 1, 'mime_type':
'MemoryMimeType.TEXT', 'score': 0.5185262858867645, 'id': 'b651bc91-3e80-4733-
bf9e-4ab17fb70c7d'})), MemoryContent(content='How frequently should progress
updates and communication occur with stakeholders? \n \nWhat are the preferred
communication channels and formats for sharing project updates \nand findings? \n
\nBudget and Resources: \n \nWhat is the allocated budget for this project? Are

```

```

there any budgetary constraints or \nconsiderations? \n \nWhat resources (e.g.,
personnel, equipment, funding) are available or needed to support \nthis project?
\n \nRisk Management:', mime_type='MemoryMimeType.TEXT', metadata={'mime_type':
'MemoryMimeType.TEXT', 'chunk_index': 3, 'source': 'Business Analyst questions
for functional requirements interviews.pdf', 'score': 0.5155279040336609, 'id':
'42ac3b50-b9d8-4528-a502-6e48caf19b74'}})]
----- ModelClientStreamingChunkEvent (interviewer_agent) -----
QUESTION: Should we document specific assumptions about resource availability
(e.g., marketing or IT teams) or technical limitations like call infrastructure
or database scalability?
----- TextMessage (user) -----
no
id='b9d21e44-5b43-4d33-871b-cf9dcb2327b1' source='interviewer_agent'
models_usage=RequestUsage(prompt_tokens=0, completion_tokens=0) metadata={}
created_at=datetime.datetime(2025, 8, 9, 20, 47, 31, 997364,
tzinfo=datetime.timezone.utc)
content=[FunctionCall(id='call_Ja4uECi9mAUfeaP3oNsvrAYG',
arguments='{"speaker_or_records":[{"speaker":"interviewer","message":"QUESTION:
Are there any assumptions about resource availability, stakeholder
responsibilities, or system limitations we should
document?","subject":"Constraints &
Assumptions","q_index":2},{ "speaker":"user","message":"no","subject":"Constraints
&
Assumptions","q_index":2}],"file_path":"../conversation/interview.jsonl","session
_id":"sess-20231011162620"}', name='save_conversation_to_jsonl'))]
type='ToolCallRequestEvent'
id='b9d21e44-5b43-4d33-871b-cf9dcb2327b1' source='interviewer_agent'
models_usage=RequestUsage(prompt_tokens=0, completion_tokens=0) metadata={}
created_at=datetime.datetime(2025, 8, 9, 20, 47, 31, 997364,
tzinfo=datetime.timezone.utc)
content=[FunctionCall(id='call_Ja4uECi9mAUfeaP3oNsvrAYG',
arguments='{"speaker_or_records":[{"speaker":"interviewer","message":"QUESTION:
Are there any assumptions about resource availability, stakeholder
responsibilities, or system limitations we should
document?","subject":"Constraints &
Assumptions","q_index":2},{ "speaker":"user","message":"no","subject":"Constraints
&
Assumptions","q_index":2}],"file_path":"../conversation/interview.jsonl","session
_id":"sess-20231011162620"}', name='save_conversation_to_jsonl'))]
type='ToolCallRequestEvent'
DEBUG:autogen_agentchat.events:id='b9d21e44-5b43-4d33-871b-cf9dcb2327b1'
source='interviewer_agent' models_usage=RequestUsage(prompt_tokens=0,
completion_tokens=0) metadata={} created_at=datetime.datetime(2025, 8, 9, 20, 47,
31, 997364, tzinfo=datetime.timezone.utc)
content=[FunctionCall(id='call_Ja4uECi9mAUfeaP3oNsvrAYG',

```

```

arguments={'speaker_or_records': [{"speaker": "interviewer", "message": "QUESTION:
Are there any assumptions about resource availability, stakeholder
responsibilities, or system limitations we should
document?", "subject": "Constraints &
Assumptions", "q_index": 2}], {"speaker": "user", "message": "no", "subject": "Constraints
&
Assumptions", "q_index": 2}], "file_path": "../conversation/interview.jsonl", "session
_id": "sess-20231011162620"}, name='save_conversation_to_jsonl')]
type='ToolCallRequestEvent'
----- ToolCallRequestEvent (interviewer_agent) -----
[FunctionCall(id='call_Ja4uECi9mAUfeaP3oNsvrAYG',
arguments={'speaker_or_records': [{"speaker": "interviewer", "message": "QUESTION:
Are there any assumptions about resource availability, stakeholder
responsibilities, or system limitations we should
document?", "subject": "Constraints &
Assumptions", "q_index": 2}], {"speaker": "user", "message": "no", "subject": "Constraints
&
Assumptions", "q_index": 2}], "file_path": "../conversation/interview.jsonl", "session
_id": "sess-20231011162620"}, name='save_conversation_to_jsonl')]
id='41871f49-a085-448a-8db0-2537b9045eb2' source='interviewer_agent'
models_usage=None metadata={} created_at=datetime.datetime(2025, 8, 9, 20, 47,
32, 1103, tzinfo=datetime.timezone.utc)
content=[FunctionExecutionResult(content='Saved 2 record(s) to
../conversation/interview.jsonl', name='save_conversation_to_jsonl',
call_id='call_Ja4uECi9mAUfeaP3oNsvrAYG', is_error=False)]
type='ToolCallExecutionEvent'
id='41871f49-a085-448a-8db0-2537b9045eb2' source='interviewer_agent'
models_usage=None metadata={} created_at=datetime.datetime(2025, 8, 9, 20, 47,
32, 1103, tzinfo=datetime.timezone.utc)
content=[FunctionExecutionResult(content='Saved 2 record(s) to
../conversation/interview.jsonl', name='save_conversation_to_jsonl',
call_id='call_Ja4uECi9mAUfeaP3oNsvrAYG', is_error=False)]
type='ToolCallExecutionEvent'
DEBUG:autogen_agentchat.events:id='41871f49-a085-448a-8db0-2537b9045eb2'
source='interviewer_agent' models_usage=None metadata={}
created_at=datetime.datetime(2025, 8, 9, 20, 47, 32, 1103,
tzinfo=datetime.timezone.utc) content=[FunctionExecutionResult(content='Saved 2
record(s) to ../conversation/interview.jsonl', name='save_conversation_to_jsonl',
call_id='call_Ja4uECi9mAUfeaP3oNsvrAYG', is_error=False)]
type='ToolCallExecutionEvent'
----- ToolCallExecutionEvent (interviewer_agent) -----
[FunctionExecutionResult(content='Saved 2 record(s) to
../conversation/interview.jsonl', name='save_conversation_to_jsonl',
call_id='call_Ja4uECi9mAUfeaP3oNsvrAYG', is_error=False)]
----- ModelClientStreamingChunkEvent (interviewer_agent) -----

```

```

Product Overview asked=2, status=done
KPI & Success Metrics asked=2, status=done
Scope In and Out asked=2, status=done
Non-Functional Requirements asked=2, status=done
User Roles & Personas asked=2, status=done
Integrations & External Systems asked=2, status=done
Constraints & Assumptions asked=2, status=done
Dependencies & Risks asked=0, status=not started

QUESTION: What are the potential risks or dependencies we should be aware of for
this project?
    response = await self._model_client.create(messages=select_speaker_messages)
Model selected a valid name: planning_agent (attempt 1)
DEBUG:autogen_agentchat:Model selected a valid name: planning_agent (attempt 1)
Selected speaker: planning_agent
DEBUG:autogen_agentchat:Selected speaker: planning_agent
----- ModelClientStreamingChunkEvent (planning_agent) -----
1 user : Answer the interviewer_agent's question about potential risks or
dependencies for this project.

[ASSIGNMENT] agent=user task="Answer the interviewer_agent's question about
potential risks or dependencies for this project." awaiting_user_reply=true

```

Sumarizer and functional_spec writer:

```

DEBUG:autogen_agentchat:Selected speaker: summarizer_agent
----- ModelClientStreamingChunkEvent (summarizer_agent) -----
The requirements and scope summary for the AI dialer were already presented and
are awaiting approval for further progression. Please confirm whether you would
like to modify this summary, proceed to finalize it, or specify additional
instructions.
To finalize, simply respond with **APPROVE SUMMARY**.
    response = await self._model_client.create(messages=select_speaker_messages)
Model selected a valid name: functional_spec_writer_agent (attempt 1)
DEBUG:autogen_agentchat:Model selected a valid name: functional_spec_writer_agent
(attempt 1)
Selected speaker: functional_spec_writer_agent
DEBUG:autogen_agentchat:Selected speaker: functional_spec_writer_agent
----- MemoryQueryEvent (functional_spec_writer_agent) -----
[MemoryContent(content='and printing requirements templates.Demo Video 8.
RequirementsWatch this video on YouTube\xa0The analyst can talk you through their
steps, techniques, tools, and processes for project completion. Their process may
look something like this:Meet with stakeholdersGather and document
requirementsGet approvalMonitor progress18. What Requirements Gathering Tool or

```

Use_case_agent:

```
Software Do You Use for Requirements Analysis?Requiment is an automated tool that
lets analysts generate documents, wireframes, and task',
mime_type='MemoryMimeType.TEXT', metadata={'source':
'https://www.requiment.com/business-analyst-requirements-gathering-interview-
questions-answers/', 'chunk_index': 28, 'mime_type': 'MemoryMimeType.TEXT',
'score': 0.4198424816131592, 'id': 'e5855e4e-36ea-4436-ad7c-a32f564a6f56'}}]
id='c0ff6789-d696-4f80-a991-889d30eae33b' source='functional_spec_writer_agent'
models_usage=RequestUsage(prompt_tokens=0, completion_tokens=0) metadata={}
created_at=datetime.datetime(2025, 8, 9, 20, 53, 32, 720088,
tzinfo=datetime.timezone.utc)
content=[FunctionCall(id='call_Y3Xs7n5K6RYiFW7Cps49EhwI',
arguments='{"task":"Generate detailed Use Cases and User Stories for the AI
Dialer system, focusing on call functionalities, reporting, database
interactions, and user roles such as marketing teams, IT support, human agents,
and customers."}', name='use_case_agent')] type='ToolCallRequestEvent'
id='c0ff6789-d696-4f80-a991-889d30eae33b' source='functional_spec_writer_agent'
models_usage=RequestUsage(prompt_tokens=0, completion_tokens=0) metadata={}
created_at=datetime.datetime(2025, 8, 9, 20, 53, 32, 720088,
tzinfo=datetime.timezone.utc)
content=[FunctionCall(id='call_Y3Xs7n5K6RYiFW7Cps49EhwI',
arguments='{"task":"Generate detailed Use Cases and User Stories for the AI
Dialer system, focusing on call functionalities, reporting, database
interactions, and user roles such as marketing teams, IT support, human agents,
and customers."}', name='use_case_agent')] type='ToolCallRequestEvent'
DEBUG:autogen_agentchat.events:id='c0ff6789-d696-4f80-a991-889d30eae33b'
source='functional_spec_writer_agent' models_usage=RequestUsage(prompt_tokens=0,
completion_tokens=0) metadata={} created_at=datetime.datetime(2025, 8, 9, 20, 53,
32, 720088, tzinfo=datetime.timezone.utc)
content=[FunctionCall(id='call_Y3Xs7n5K6RYiFW7Cps49EhwI',
arguments='{"task":"Generate detailed Use Cases and User Stories for the AI
Dialer system, focusing on call functionalities, reporting, database
interactions, and user roles such as marketing teams, IT support, human agents,
and customers."}', name='use_case_agent')] type='ToolCallRequestEvent'
----- ToolCallRequestEvent (functional_spec_writer_agent) -----
[FunctionCall(id='call_Y3Xs7n5K6RYiFW7Cps49EhwI', arguments='{"task":"Generate
detailed Use Cases and User Stories for the AI Dialer system, focusing on call
functionalities, reporting, database interactions, and user roles such as
marketing teams, IT support, human agents, and customers."}',
name='use_case_agent')]
----- TextMessage (user) -----
Generate detailed Use Cases and User Stories for the AI Dialer system, focusing
on call functionalities, reporting, database interactions, and user roles such as
marketing teams, IT support, human agents, and customers.
```

```
----- MemoryQueryEvent (use_case_agent) -----  
[MemoryContent(content='to achieve specific goals, and user stories refer to  
descriptions of what users want from your product. Finally, acceptance criteria  
are the final requirements to be met before a project is complete.21. How Do You  
Design a Use Case?You want an analyst who can explain his step-by-step process  
for use cases because they require structure. Use cases are critical to your  
software's user experience. Analysts design two use cases: one is a business use  
case and the other is a system use case. Ask', mime_type='MemoryMimeType.TEXT',  
metadata={'chunk_index': 31, 'mime_type': 'MemoryMimeType.TEXT', 'source':  
'https://www.requiment.com/business-analyst-requirements-gathering-interview-  
questions-answers/', 'score': 0.481245756149292, 'id': 'ce6a2585-6c5a-4067-bcd8-  
4c529336398c'}), MemoryContent(content='Understanding the nuances of use case  
flows showcases the candidate's attention to detail and their ability to handle  
different scenarios.\n33. What Are Personas and How Are They Useful in User-  
Centered Design?\nPersonas are fictional characters representing user types. This  
question assesses the candidate's understanding of user-centric design  
principles.\n34. How Would You Define Analytical Reporting?',  
mime_type='MemoryMimeType.TEXT', metadata={'chunk_index': 47, 'mime_type':  
'MemoryMimeType.TEXT', 'source': 'https://www.morebusiness.com/business-analyst-  
interview-questions/', 'score': 0.4273034334182739, 'id': '03986690-066d-45fa-  
82c2-7b72c0c51d90'}), MemoryContent(content='The Safest Easiest App for Authentic  
AI Content Creation \\\n\\\n\\\n\\\nGet More Business \\\n\\\n\\\n\\\n\\\n\\\n\\\n\\\n\\\n\\\nMake  
Content\\\n\\\n\\\n\\\n\\\n\\\n\\\n\\\n\\\n\\\n\\\n\\\n\\\n\\\n\\\nMarketing Tools \\\n\\\n\\\n\\\n\\\n\\\n\\\nSafe Authentic AI Content Creation\\\n\\\n\\\n\\\n\\\n\\\nSEO Keyword Tool\\\n\\\n\\\n\\\n\\\n\\\nMSP Website Content Kit\\\n\\\n\\\n\\\n\\\n\\\nDone-for-You Content\\\n\\\n\\\n\\\n\\\n\\\nLogo  
Maker\\\n\\\n\\\n\\\n\\\nGraphic Design Tool\\\n\\\n\\\n\\\n\\\n\\\nHubspot Marketing  
Automation\\\n\\\n\\\n\\\n\\\n\\\n\\\n\\\n\\\nPopular Downloads \\\n\\\n\\\n\\\n\\\n\\\nMarketing  
Plan Example', mime_type='MemoryMimeType.TEXT', metadata={'mime_type':  
'MemoryMimeType.TEXT', 'chunk_index': 79, 'source':  
'https://www.morebusiness.com/business-analyst-interview-questions/', 'score':  
0.42291587591171265, 'id': 'd1f43a19-236e-4a51-9b47-01a7da6a56df'})]  
----- ModelClientStreamingChunkEvent (use_case_agent) -----  
# Use Cases and User Stories for AI Dialer System  
  
## Use Cases  
  
### **Use Case 1: Make an Outbound Call**  
#### Description:  
The AI Dialer enables human agents to make outbound calls efficiently to  
customers, providing behavioral insights to tailor the call experience.  
  
#### Actors:  
- Human Agents  
- Customers
```

Preconditions:

1. The agent must be logged into the AI Dialer system.
2. The customer's contact information must exist in the database.
3. The outbound call feature must be enabled for the agent.

Diagrams and Agent task completed:

```
DEBUG:autogen_agentchat:Selected speaker: diagram_creator_agent
----- MemoryQueryEvent (diagram_creator_agent) -----
[MemoryContent(content='=== CANVAS FILES ===\n=== END OF CANVAS ===',
mime_type=<MemoryMimeType.TEXT: 'text/plain'>, metadata=None)]
id='1629d835-e444-48e9-906c-cb0ff90ad734' source='diagram_creator_agent'
models_usage=RequestUsage(prompt_tokens=0, completion_tokens=0) metadata={}
created_at=datetime.datetime(2025, 8, 9, 20, 59, 1, 80974,
tzinfo=datetime.timezone.utc)
content=[FunctionCall(id='call_20bUiidTRVjsUWZR3i09wuwr',
arguments='{"filename":"canvas_diagram.md","new_content":"%% Functional
Specification Diagram\\n\\nflowchart TD\\n    Start[\\\"Start\\\"] -->
AI_Dialer\\n    AI_Dialer[\\\"AI Dialer Initialization\\\"] --> |Fetch customer
data from CRM| FetchData{\\\"Fetch Data\\\"}\\n    FetchData --> |Fetch successful|
VerifyHours[\\\"Verify call time is within 8 AM to 5 PM\\\"]\\n    VerifyHours -->
|Within allowed hours| MakeCall[\\\"Initiate Call\\\"]\\n    MakeCall --> |Call
connected| HandleInteraction[\\\"Handle Customer Interaction\\\"]\\n    MakeCall --
> |Call fails| RetryCall[\\\"Retry Call (Up to 3 times)\\\"]\\n    RetryCall -->
|Retries exhausted| RecordFail[\\\"Record Call as Failed\\\"]\\n    RecordFail -->
End[\\\"End\\\"]\\n    HandleInteraction --> |Customer accepts offer|
TransferToAgent[\\\"Transfer to Human Agent\\\"]\\n    HandleInteraction -->
|Customer declines repeatedly| RecordDecline[\\\"Record Declined
Deal\\\"]\\n    HandleInteraction --> |Customer has doubts/questions|
UseRAG[\\\"Consult RAG System\\\"]\\n    TransferToAgent -->
End\\n    RecordDecline --> End\\n    UseRAG -->
HandleInteraction\\n    End[\\\"End\\\"]"}', name='update_file')]
type='ToolCallRequestEvent'
id='1629d835-e444-48e9-906c-cb0ff90ad734' source='diagram_creator_agent'
models_usage=RequestUsage(prompt_tokens=0, completion_tokens=0) metadata={}
created_at=datetime.datetime(2025, 8, 9, 20, 59, 1, 80974,
tzinfo=datetime.timezone.utc)
content=[FunctionCall(id='call_20bUiidTRVjsUWZR3i09wuwr',
arguments='{"filename":"canvas_diagram.md","new_content":"%% Functional
Specification Diagram\\n\\nflowchart TD\\n    Start[\\\"Start\\\"] -->
AI_Dialer\\n    AI_Dialer[\\\"AI Dialer Initialization\\\"] --> |Fetch customer
data from CRM| FetchData{\\\"Fetch Data\\\"}\\n    FetchData --> |Fetch successful|
VerifyHours[\\\"Verify call time is within 8 AM to 5 PM\\\"]\\n    VerifyHours -->
```

```

|Within allowed hours| MakeCall[\\\"Initiate Call\\\"]\\n    MakeCall --> |Call
connected| HandleInteraction[\\\"Handle Customer Interaction\\\"]\\n    MakeCall --
> |Call fails| RetryCall[\\\"Retry Call (Up to 3 times)\\\"]\\n    RetryCall -->
|Retries exhausted| RecordFail[\\\"Record Call as Failed\\\"]\\n    RecordFail -->
End[\\\"End\\\"]\\n    HandleInteraction --> |Customer accepts offer|
TransferToAgent[\\\"Transfer to Human Agent\\\"]\\n    HandleInteraction -->
|Customer declines repeatedly| RecordDecline[\\\"Record Declined
Deal\\\"]\\n    HandleInteraction --> |Customer has doubts/questions|
UseRAG[\\\"Consult RAG System\\\"]\\n    TransferToAgent -->
End\\n    RecordDecline --> End\\n    UseRAG -->
HandleInteraction\\n    End[\\\"End\\\"]}', name='update_file')]
type='ToolCallRequestEvent'
DEBUG:autogen_agentchat.events:id='1629d835-e444-48e9-906c-cb0ff90ad734'
source='diagram_creator_agent' models_usage=RequestUsage(prompt_tokens=0,
completion_tokens=0) metadata={} created_at=datetime.datetime(2025, 8, 9, 20, 59,
1, 80974, tzinfo=datetime.timezone.utc)
content=[FunctionCall(id='call_20bUiiDTRVjsUWZR3i09wuWR',
arguments='{\"filename\":\"canvas_diagram.md\",\"new_content\":\"%% Functional
Specification Diagram\\n\\nflowchart TD\\n    Start[\\\"Start\\\"] -->
AI_Dialer\\n    AI_Dialer[\\\"AI Dialer Initialization\\\"] --> |Fetch customer
data from CRM| FetchData{\\\"Fetch Data\\\"}\\n    FetchData --> |Fetch successful|
VerifyHours[\\\"Verify call time is within 8 AM to 5 PM\\\"]\\n    VerifyHours -->
|Within allowed hours| MakeCall[\\\"Initiate Call\\\"]\\n    MakeCall --> |Call
connected| HandleInteraction[\\\"Handle Customer Interaction\\\"]\\n    MakeCall --
> |Call fails| RetryCall[\\\"Retry Call (Up to 3 times)\\\"]\\n    RetryCall -->
|Retries exhausted| RecordFail[\\\"Record Call as Failed\\\"]\\n    RecordFail -->
End[\\\"End\\\"]\\n    HandleInteraction --> |Customer accepts offer|
TransferToAgent[\\\"Transfer to Human Agent\\\"]\\n    HandleInteraction -->
|Customer declines repeatedly| RecordDecline[\\\"Record Declined
Deal\\\"]\\n    HandleInteraction --> |Customer has doubts/questions|
UseRAG[\\\"Consult RAG System\\\"]\\n    TransferToAgent -->
End\\n    RecordDecline --> End\\n    UseRAG -->
HandleInteraction\\n    End[\\\"End\\\"]}', name='update_file')]
type='ToolCallRequestEvent'
----- ToolCallRequestEvent (diagram_creator_agent) -----
id='2a0388c7-bb5a-42f9-80f8-198fee9cdab7' source='diagram_creator_agent'
models_usage=None metadata={} created_at=datetime.datetime(2025, 8, 9, 20, 59, 1,
83554, tzinfo=datetime.timezone.utc)
content=[FunctionExecutionResult(content='{\"status\": \"OK\"}', name='update_file',
call_id='call_20bUiiDTRVjsUWZR3i09wuWR', is_error=False)]
type='ToolCallExecutionEvent'
id='2a0388c7-bb5a-42f9-80f8-198fee9cdab7' source='diagram_creator_agent'
models_usage=None metadata={} created_at=datetime.datetime(2025, 8, 9, 20, 59, 1,
83554, tzinfo=datetime.timezone.utc)
content=[FunctionExecutionResult(content='{\"status\": \"OK\"}', name='update_file',

```

```

call_id='call_20bUiIDTRVjsUWZR3i09wuWR', is_error=False)]
type='ToolCallExecutionEvent'
DEBUG:autogen_agentchat.events:id='2a0388c7-bb5a-42f9-80f8-198fee9cdab7'
source='diagram_creator_agent' models_usage=None metadata={}
created_at=datetime.datetime(2025, 8, 9, 20, 59, 1, 83554,
tzinfo=datetime.timezone.utc)
content=[FunctionExecutionResult(content='{"status": "OK"}', name='update_file',
call_id='call_20bUiIDTRVjsUWZR3i09wuWR', is_error=False)]
type='ToolCallExecutionEvent'
[FunctionCall(id='call_20bUiIDTRVjsUWZR3i09wuWR',
arguments={'filename':"canvas_diagram.md","new_content":"%% Functional
Specification Diagram\\n\\nflowchart TD\\n    Start[\\\"Start\\\"] -->
AI_Dialer\\n    AI_Dialer[\\\"AI Dialer Initialization\\\"] --> |Fetch customer
data from CRM| FetchData{\\\"Fetch Data\\\"}\\n    FetchData --> |Fetch successful|
VerifyHours[\\\"Verify call time is within 8 AM to 5 PM\\\"]\\n    VerifyHours -->
|Within allowed hours| MakeCall[\\\"Initiate Call\\\"]\\n    MakeCall --> |Call
connected| HandleInteraction[\\\"Handle Customer Interaction\\\"]\\n    MakeCall --
> |Call fails| RetryCall[\\\"Retry Call (Up to 3 times)\\\"]\\n    RetryCall -->
|Retries exhausted| RecordFail[\\\"Record Call as Failed\\\"]\\n    RecordFail -->
End[\\\"End\\\"]\\n    HandleInteraction --> |Customer accepts offer|
TransferToAgent[\\\"Transfer to Human Agent\\\"]\\n    HandleInteraction -->
|Customer declines repeatedly| RecordDecline[\\\"Record Declined
Deal\\\"]\\n    HandleInteraction --> |Customer has doubts/questions|
UseRAG[\\\"Consult RAG System\\\"]\\n    TransferToAgent -->
End\\n    RecordDecline --> End\\n    UseRAG -->
HandleInteraction\\n    End[\\\"End\\\"]"}', name='update_file')]
----- ToolCallExecutionEvent (diagram_creator_agent) -----
[FunctionExecutionResult(content='{"status": "OK"}', name='update_file',
call_id='call_20bUiIDTRVjsUWZR3i09wuWR', is_error=False)]
----- ModelClientStreamingChunkEvent (diagram_creator_agent) -----
Diagram code complete. Handing off to mermaid_code_reviewer_agent.
    response = await self._model_client.create(messages=select_speaker_messages)
Model selected a valid name: mermaid_code_reviewer_agent (attempt 1)
DEBUG:autogen_agentchat:Model selected a valid name: mermaid_code_reviewer_agent
(attempt 1)
Selected speaker: mermaid_code_reviewer_agent
DEBUG:autogen_agentchat:Selected speaker: mermaid_code_reviewer_agent
----- MemoryQueryEvent (mermaid_code_reviewer_agent) -----
[MemoryContent(content='=== CANVAS FILES ===\\nFile: canvas_diagram.md (rev
1):\\n%% Functional Specification Diagram\\n\\nflowchart TD\\n    Start["Start"] -->
AI_Dialer\\n    AI_Dialer["AI Dialer Initialization"] --> |Fetch customer
data from CRM| FetchData{"Fetch Data"}\\n    FetchData --> |Fetch successful|
VerifyHours["Verify call time is within 8 AM to 5 PM"]\\n    VerifyHours -->
|Within allowed hours| MakeCall["Initiate Call"]\\n    MakeCall --> |Call
connected| HandleInteraction["Handle Customer Interaction"]\\n    MakeCall -->

```

```

|Call fails| RetryCall["Retry Call (Up to 3 times)"]\n    RetryCall --> |Retries
exhausted| RecordFail["Record Call as Failed"]\n    RecordFail -->
End["End"]\n    HandleInteraction --> |Customer accepts offer|
TransferToAgent["Transfer to Human Agent"]\n    HandleInteraction --> |Customer
declines repeatedly| RecordDecline["Record Declined Deal"]\n    HandleInteraction
--> |Customer has doubts/questions| UseRAG["Consult RAG
System"]\n    TransferToAgent --> End\n    RecordDecline --> End\n    UseRAG -->
HandleInteraction\n    End["End"]\n\n=== END OF CANVAS ===',
mime_type=<MemoryMimeType.TEXT: 'text/plain'>, metadata=None)]
id='8be571fd-2ded-4090-885b-6314f191e4b4' source='mermaid_code_reviewer_agent'
models_usage=RequestUsage(prompt_tokens=0, completion_tokens=0) metadata={}
created_at=datetime.datetime(2025, 8, 9, 20, 59, 41, 831534,
tzinfo=datetime.timezone.utc)
content=[FunctionCall(id='call_ULZNs9eJLGH9HR0Igxgo8MVz',
arguments={'mermaid_code':"flowchart TD\n    Start[\"Start\"] -->
AI_Dialer\n    AI_Dialer[\"AI Dialer Initialization\"] --> |Fetch customer
data from CRM| FetchData[\"Fetch Data\"]\n    FetchData --> |Fetch successful|
VerifyHours[\"Verify call time is within 8 AM to 5 PM\"]\n    VerifyHours -->
|Within allowed hours| MakeCall[\"Initiate Call\"]\n    MakeCall --> |Call
connected| HandleInteraction[\"Handle Customer Interaction\"]\n    MakeCall --
> |Call fails| RetryCall[\"Retry Call (Up to 3 times)\"]\n    RetryCall -->
|Retries exhausted| RecordFail[\"Record Call as Failed\"]\n    RecordFail -->
End[\"End\"]\n    HandleInteraction --> |Customer accepts offer|
TransferToAgent[\"Transfer to Human Agent\"]\n    HandleInteraction -->
|Customer declines repeatedly| RecordDecline[\"Record Declined
Deal\"]\n    HandleInteraction --> |Customer has doubts/questions|
UseRAG[\"Consult RAG System\"]\n    TransferToAgent -->
End\n    RecordDecline --> End\n    UseRAG -->
HandleInteraction\n    End[\"End\"]"}', name='write_mermaid_to_file_tool')]
type='ToolCallRequestEvent'
id='8be571fd-2ded-4090-885b-6314f191e4b4' source='mermaid_code_reviewer_agent'
models_usage=RequestUsage(prompt_tokens=0, completion_tokens=0) metadata={}
created_at=datetime.datetime(2025, 8, 9, 20, 59, 41, 831534,
tzinfo=datetime.timezone.utc)
content=[FunctionCall(id='call_ULZNs9eJLGH9HR0Igxgo8MVz',
arguments={'mermaid_code':"flowchart TD\n    Start[\"Start\"] -->
AI_Dialer\n    AI_Dialer[\"AI Dialer Initialization\"] --> |Fetch customer
data from CRM| FetchData[\"Fetch Data\"]\n    FetchData --> |Fetch successful|
VerifyHours[\"Verify call time is within 8 AM to 5 PM\"]\n    VerifyHours -->
|Within allowed hours| MakeCall[\"Initiate Call\"]\n    MakeCall --> |Call
connected| HandleInteraction[\"Handle Customer Interaction\"]\n    MakeCall --
> |Call fails| RetryCall[\"Retry Call (Up to 3 times)\"]\n    RetryCall -->
|Retries exhausted| RecordFail[\"Record Call as Failed\"]\n    RecordFail -->
End[\"End\"]\n    HandleInteraction --> |Customer accepts offer|
TransferToAgent[\"Transfer to Human Agent\"]\n    HandleInteraction -->

```

```

|Customer declines repeatedly| RecordDecline[\\\"Record Declined
Deal\\\"]\\n    HandleInteraction --> |Customer has doubts/questions|
UseRAG[\\\"Consult RAG System\\\"]\\n    TransferToAgent -->
End\\n    RecordDecline --> End\\n    UseRAG -->
HandleInteraction\\n    End[\\\"End\\\"]}\"', name='write_mermaid_to_file_tool')]
type='ToolCallRequestEvent'
DEBUG:autogen_agentchat.events:id='8be571fd-2ded-4090-885b-6314f191e4b4'
source='mermaid_code_reviewer_agent' models_usage=RequestUsage(prompt_tokens=0,
completion_tokens=0) metadata={} created_at=datetime.datetime(2025, 8, 9, 20, 59,
41, 831534, tzinfo=datetime.timezone.utc)
content=[FunctionCall(id='call_ULZNs9eJLGH9HR0Igxgo8MVz',
arguments='{\"mermaid_code\":\"flowchart TD\\n    Start[\\\"Start\\\"] -->
AI_Dialer\\n    AI_Dialer[\\\"AI Dialer Initialization\\\"] --> |Fetch customer
data from CRM| FetchData{\\\"Fetch Data\\\"}\\n    FetchData --> |Fetch successful|
VerifyHours[\\\"Verify call time is within 8 AM to 5 PM\\\"]\\n    VerifyHours -->
|Within allowed hours| MakeCall[\\\"Initiate Call\\\"]\\n    MakeCall --> |Call
connected| HandleInteraction[\\\"Handle Customer Interaction\\\"]\\n    MakeCall --
> |Call fails| RetryCall[\\\"Retry Call (Up to 3 times)\\\"]\\n    RetryCall -->
|Retries exhausted| RecordFail[\\\"Record Call as Failed\\\"]\\n    RecordFail -->
End[\\\"End\\\"]\\n    HandleInteraction --> |Customer accepts offer|
TransferToAgent[\\\"Transfer to Human Agent\\\"]\\n    HandleInteraction -->
|Customer declines repeatedly| RecordDecline[\\\"Record Declined
Deal\\\"]\\n    HandleInteraction --> |Customer has doubts/questions|
UseRAG[\\\"Consult RAG System\\\"]\\n    TransferToAgent -->
End\\n    RecordDecline --> End\\n    UseRAG -->
HandleInteraction\\n    End[\\\"End\\\"]}\"', name='write_mermaid_to_file_tool')]
type='ToolCallRequestEvent'
----- ToolCallRequestEvent (mermaid_code_reviewer_agent) -----
[FunctionCall(id='call_ULZNs9eJLGH9HR0Igxgo8MVz',
arguments='{\"mermaid_code\":\"flowchart TD\\n    Start[\\\"Start\\\"] -->
AI_Dialer\\n    AI_Dialer[\\\"AI Dialer Initialization\\\"] --> |Fetch customer
data from CRM| FetchData{\\\"Fetch Data\\\"}\\n    FetchData --> |Fetch successful|
VerifyHours[\\\"Verify call time is within 8 AM to 5 PM\\\"]\\n    VerifyHours -->
|Within allowed hours| MakeCall[\\\"Initiate Call\\\"]\\n    MakeCall --> |Call
connected| HandleInteraction[\\\"Handle Customer Interaction\\\"]\\n    MakeCall --
> |Call fails| RetryCall[\\\"Retry Call (Up to 3 times)\\\"]\\n    RetryCall -->
|Retries exhausted| RecordFail[\\\"Record Call as Failed\\\"]\\n    RecordFail -->
End[\\\"End\\\"]\\n    HandleInteraction --> |Customer accepts offer|
TransferToAgent[\\\"Transfer to Human Agent\\\"]\\n    HandleInteraction -->
|Customer declines repeatedly| RecordDecline[\\\"Record Declined
Deal\\\"]\\n    HandleInteraction --> |Customer has doubts/questions|
UseRAG[\\\"Consult RAG System\\\"]\\n    TransferToAgent -->
End\\n    RecordDecline --> End\\n    UseRAG -->
HandleInteraction\\n    End[\\\"End\\\"]}\"', name='write_mermaid_to_file_tool')]

```

```

id='5e9d8c07-786c-4718-aacf-676701f51d0a' source='mermaid_code_reviewer_agent'
models_usage=None metadata={} created_at=datetime.datetime(2025, 8, 9, 20, 59,
41, 834798, tzinfo=datetime.timezone.utc)
content=[FunctionExecutionResult(content='Mermaid diagram successfully written to
../conversation/process_diagram.mmd', name='write_mermaid_to_file_tool',
call_id='call_ULZNs9eJLGH9HR0Igxgo8MVz', is_error=False)]
type='ToolCallExecutionEvent'
id='5e9d8c07-786c-4718-aacf-676701f51d0a' source='mermaid_code_reviewer_agent'
models_usage=None metadata={} created_at=datetime.datetime(2025, 8, 9, 20, 59,
41, 834798, tzinfo=datetime.timezone.utc)
content=[FunctionExecutionResult(content='Mermaid diagram successfully written to
../conversation/process_diagram.mmd', name='write_mermaid_to_file_tool',
call_id='call_ULZNs9eJLGH9HR0Igxgo8MVz', is_error=False)]
type='ToolCallExecutionEvent'
DEBUG:autogen_agentchat.events:id='5e9d8c07-786c-4718-aacf-676701f51d0a'
source='mermaid_code_reviewer_agent' models_usage=None metadata={}
created_at=datetime.datetime(2025, 8, 9, 20, 59, 41, 834798,
tzinfo=datetime.timezone.utc) content=[FunctionExecutionResult(content='Mermaid
diagram successfully written to ../conversation/process_diagram.mmd',
name='write_mermaid_to_file_tool', call_id='call_ULZNs9eJLGH9HR0Igxgo8MVz',
is_error=False)] type='ToolCallExecutionEvent'
----- ToolCallExecutionEvent (mermaid_code_reviewer_agent) -----
[FunctionExecutionResult(content='Mermaid diagram successfully written to
../conversation/process_diagram.mmd', name='write_mermaid_to_file_tool',
call_id='call_ULZNs9eJLGH9HR0Igxgo8MVz', is_error=False)]
----- ModelClientStreamingChunkEvent (mermaid_code_reviewer_agent) -----
-
Diagram code complete
**READY**
    response = await self._model_client.create(messages=select_speaker_messages)
Model selected a valid name: planning_agent (attempt 1)
DEBUG:autogen_agentchat:Model selected a valid name: planning_agent (attempt 1)
Selected speaker: planning_agent
DEBUG:autogen_agentchat:Selected speaker: planning_agent
----- ModelClientStreamingChunkEvent (planning_agent) -----
All tasks across the agents have successfully been completed. Here's a summary of
the work performed:

### Summary:
1. **Interviewer_agent**: Gathered user requirements and scoped the AI dialer
project. Successfully identified functionalities, integrations, KPIs,
constraints, and dependencies.
2. **Summarizer_agent**: Consolidated all gathered information into a coherent
summary, which was reviewed and approved.

```

```

3. **Functional_spec_writer_agent**: Drafted the functional specification document, including the key sections.
4. **Functional_spec_checker_agent**: Reviewed the specification document for completeness and approved it.
5. **Diagram_creator_agent**: Created process flow diagrams to visually represent the AI dialer system.
6. **Mermaid_code_reviewer_agent**: Reviewed the code for the diagrams for clarity and correctness, confirming readiness.

```

Terminate_agent:

```

Execute the termination pipeline
----- ToolCallRequestEvent (terminate_agent) -----
[FunctionCall(id='call_crcgWvONfwnhyp66ILOj8F2v',
arguments={'path': "/run/media/jun/Jun_6T/Code/TFM_BA_GPT/conversation/interview.jsonl"}, name='read_file')]
----- ToolCallExecutionEvent (terminate_agent) -----
[FunctionExecutionResult(content='[{"type": "text", "text": "{\\"_meta\\": {\\"session_id\\": \\"sess-20231011162620\\", \\"ts\\": \\"2025-08-09T20:35:33.157973Z\\", \\"n_records\\": 2}}\\n{\\"speaker\\": \\"interviewer\\", \\"message\\": \\"QUESTION: Can you briefly describe the new product and its primary purpose?\\", \\"subject\\": \\"Product Overview\\", \\"q_index\\": 1, \\"timestamp\\": \\"2025-08-09T20:35:33.157947Z\\"}\\n{\\"speaker\\": \\"user\\", \\"message\\": \\"The new product is an AI dialer that calls customers to offer travel packages. If the customer accepts, the call transfers to a human. If declined, the AI attempts to convince the customer up to three additional times before ending the call\\", \\"subject\\": \\"Product Overview\\", \\"q_index\\": 1, \\"timestamp\\": \\"2025-08-09T20:35:33.157947Z\\"}\\n{\\"_meta\\": {\\"session_id\\": \\"sess-20231011162620\\", \\"ts\\": \\"2025-08-09T20:36:12.337556Z\\", \\"n_records\\": 2}}\\n{\\"speaker\\": \\"interviewer\\", \\"message\\": \\"QUESTION: What are the key features or functionalities that you envision for this AI dialer?\\", \\"subject\\": \\"Product Overview\\", \\"q_index\\": 2, \\"timestamp\\": \\"2025-08-09T20:36:12.337533Z\\"}\\n{\\"speaker\\": \\"user\\", \\"message\\": \\"scope in: the dialer will be integrated with the call center system to make and transfer calls, with the CRM to get customer information like name, phone number, location, salary, with a database to get the pre defined offers and to save calls result as succesful call, closed deals, rejected deals, call errors and a RAG to answer customer doubts and questions, t the end of the shift the AI shouuld make a report with metrics and KPIs, scope out: whats app integration, social network integration\\", \\"subject\\": \\"Product Overview\\", \\"q_index\\": 2, \\"timestamp\\": \\"2025-08-09T20:36:12.337533Z\\"}\\n{\\"_meta\\": {\\"session_id\\": \\"sess-20231011162620\\", \\"ts\\": \\"2025-08-09T20:36:40.339321Z\\", \\"n_records\\": 2}}\\n{\\"speaker\\": \\"interviewer\\", \\"message\\": \\"QUESTION: How would you define success for this project in terms of measurable business objectives?\\", \\"subject\\": \\"KPI & Success

```

```

Metrics\\", \\\"q_index\\\": 1, \\\"timestamp\\\": \\\"2025-08-
09T20:36:40.339292Z\\\"}\\n{\\\"speaker\\\": \\\"user\\\", \\\"message\\\": \\\"the KPI
will be configurable and metrics will include succesfull calls, error calls,
succesfull deal, rejected deals, the reprot should include chats, tables and a
dynamic dashboard will all KPI and metrics, Performance: the syetem should be
able to support up to 300 call simultanously, Reliability:have an avaiabiity of
99.96%, responses have to be in real time, and use natural language and human
like voice quality, Usability: use best prctices, Security:use access control,
dev full access to app and code, marketting to reports\\\", \\\"subject\\\": \\\"KPI
& Success Metrics\\\", \\\"q_index\\\": 1, \\\"timestamp\\\": \\\"2025-08-
09T20:36:40.339292Z\\\"}\\n{\\\"_meta\\\": {\\\"session_id\\\": \\\"sess-
20231011162620\\\", \\\"ts\\\": \\\"2025-08-09T20:37:13.210804Z\\\", \\\"n_records\\\":
2}}\\n{\\\"speaker\\\": \\\"interviewer\\\", \\\"message\\\": \\\"QUESTION: Should
additional reporting capabilities or real-time data visualization options be part
of the MVP?\\\", \\\"subject\\\": \\\"KPI & Success Metrics\\\", \\\"q_index\\\": 2,
\\\"timestamp\\\": \\\"2025-08-09T20:37:13.210784Z\\\"}\\n{\\\"speaker\\\": \\\"user\\\",
\\\"message\\\": \\\"no\\\", \\\"subject\\\": \\\"KPI & Success Metrics\\\",
\\\"q_index\\\": 2, \\\"timestamp\\\": \\\"2025-08-
09T20:37:13.210784Z\\\"}\\n{\\\"_meta\\\": {\\\"session_id\\\": \\\"sess-
20231011162620\\\", \\\"ts\\\": \\\"2025-08-09T20:37:44.574598Z\\\", \\\"n_records\\\":
2}}\\n{\\\"speaker\\\": \\\"interviewer\\\", \\\"message\\\": \\\"QUESTION: Are there
any functionalities or integrations that can be delayed for future versions?\\\",
\\\"subject\\\": \\\"Scope In and Out\\\", \\\"q_index\\\": 1, \\\"timestamp\\\":
\\\"2025-08-09T20:37:44.574581Z\\\"}\\n{\\\"speaker\\\": \\\"user\\\", \\\"message\\\":
\\\"scope in: the dialer will be integrated with the call center system to make
and transfer calls, with the CRM to get customer information like name, phone
number, location, salary, with a database to get the pre defined offers and to
save calls resulst as succesful call, closed deals, rejected deals, call errors
and a RAG to answer customer doubts and questions, t the end of the shift the AI
should make a report with metrics and KPIs\\\", \\\"subject\\\": \\\"Scope In and
Out\\\", \\\"q_index\\\": 1, \\\"timestamp\\\": \\\"2025-08-
09T20:37:44.574581Z\\\"}\\n{\\\"_meta\\\": {\\\"session_id\\\": \\\"sess-
20231011162620\\\", \\\"ts\\\": \\\"2025-08-09T20:38:20.921468Z\\\", \\\"n_records\\\":
2}}\\n{\\\"speaker\\\": \\\"interviewer\\\", \\\"message\\\": \\\"QUESTION: Are there
specific use cases or customer demographics the product should prioritize
first?\\\", \\\"subject\\\": \\\"Scope In and Out\\\", \\\"q_index\\\": 2,
\\\"timestamp\\\": \\\"2025-08-09T20:38:20.921445Z\\\"}\\n{\\\"speaker\\\": \\\"user\\\",
\\\"message\\\": \\\"That will be configurable and will depend on each campaign\\\",
\\\"subject\\\": \\\"Scope In and Out\\\", \\\"q_index\\\": 2, \\\"timestamp\\\":
\\\"2025-08-09T20:38:20.921445Z\\\"}\\n{\\\"_meta\\\": {\\\"session_id\\\": \\\"sess-
20231011162620\\\", \\\"ts\\\": \\\"2025-08-09T20:38:55.328313Z\\\", \\\"n_records\\\":
2}}\\n{\\\"speaker\\\": \\\"interviewer\\\", \\\"message\\\": \\\"QUESTION: Are there
any specific regulatory requirements or compliance standards the system must
adhere to?\\\", \\\"subject\\\": \\\"Non-Functional Requirements\\\", \\\"q_index\\\":
1, \\\"timestamp\\\": \\\"2025-08-09T20:38:55.328292Z\\\"}\\n{\\\"speaker\\\":

```

```

\\"user\\", \\"message\\": \\"Performance: the syetem should be able to support
up to 300 call simultanously, Reliability:have an avaiabiity of 99.96%,
responses have to be in real time, and use natural language and human like voice
quality, Usability: use best prctices, Security:use access control, dev full
access to app and code, marketting to reports\\", \\"subject\\": \\"Non-
Functional Requirements\\", \\"q_index\\": 1, \\"timestamp\\": \\"2025-08-
09T20:38:55.328292Z\\\"}\\n{\\\"_meta\\": {\\\"session_id\\": \\"sess-
20231011162620\\\", \\"ts\\": \\"2025-08-09T20:41:11.244248Z\\\", \\"n_records\\":
2}}\\n{\\\"speaker\\": \\"interviewer\\\", \\"message\\": \\"QUESTION: Are there
additional constraints like language support, accessibility, or anything else we
should consider?\\\", \\"subject\\": \\"Non-Functional Requirements\\\",
\\\"q_index\\": 2, \\"timestamp\\": \\"2025-08-
09T20:41:11.244225Z\\\"}\\n{\\\"speaker\\": \\"user\\\", \\"message\\": \\"it will
be only english, voice quality has to be human like level, the tone has to be
respectful but convingin\\\", \\"subject\\": \\"Non-Functional Requirements\\\",
\\\"q_index\\": 2, \\"timestamp\\": \\"2025-08-
09T20:41:11.244225Z\\\"}\\n{\\\"_meta\\": {\\\"session_id\\": \\"sess-
20231011162620\\\", \\"ts\\": \\"2025-08-09T20:41:32.082905Z\\\", \\"n_records\\":
2}}\\n{\\\"speaker\\": \\"interviewer\\\", \\"message\\": \\"QUESTION: Can you
describe the main user roles and their interactions with the AI dialer?\\\",
\\\"subject\\": \\"User Roles & Personas\\\", \\"q_index\\": 1, \\"timestamp\\":
\\\"2025-08-09T20:41:32.082886Z\\\"}\\n{\\\"speaker\\": \\"user\\\", \\"message\\":
\\\"the users wil be Travel agencies and airlines, the project stake holders are,
marketting team provide pre defined offers and pre defined answers and
documentation and past surveys for RAG, marketiing VP is the sponsor, It team for
support and development, human agents to take transfer and close deals are
testers, and the final custoemr from the companies that will receive the call\\\",
\\\"subject\\": \\"User Roles & Personas\\\", \\"q_index\\": 1, \\"timestamp\\":
\\\"2025-08-09T20:41:32.082886Z\\\"}\\n{\\\"_meta\\": {\\\"session_id\\": \\"sess-
20231011162620\\\", \\"ts\\": \\"2025-08-09T20:43:11.256210Z\\\", \\"n_records\\":
2}}\\n{\\\"speaker\\": \\"interviewer\\\", \\"message\\": \\"QUESTION: Are there
specific user personas or scenarios that need to be addressed in the system
design?\\\", \\"subject\\": \\"User Roles & Personas\\\", \\"q_index\\": 2,
\\\"timestamp\\": \\"2025-08-09T20:43:11.256187Z\\\"}\\n{\\\"speaker\\": \\"user\\\",
\\\"message\\": \\"what I already told\\\", \\"subject\\": \\"User Roles &
Personas\\\", \\"q_index\\": 2, \\"timestamp\\": \\"2025-08-
09T20:43:11.256187Z\\\"}\\n{\\\"_meta\\": {\\\"session_id\\": \\"sess-
20231011162620\\\", \\"ts\\": \\"2025-08-09T20:44:30.187823Z\\\", \\"n_records\\":
2}}\\n{\\\"speaker\\": \\"interviewer\\\", \\"message\\": \\"QUESTION: What systems
or external integrations are necessary for the AI dialer to function
effectively?\\\", \\"subject\\": \\"Integrations & External Systems\\\",
\\\"q_index\\": 1, \\"timestamp\\": \\"2025-08-
09T20:44:30.187799Z\\\"}\\n{\\\"speaker\\": \\"user\\\", \\"message\\": \\"the
dialer will be integrated with the call center system to make and transfer calls,
with the CRM to get customer information like name, phone number, location,

```

salary, with a database to get the pre defined offers and to save calls

```

resulst\\", \\\"subject\\\": \\\"Integrations & External Systems\\\", \\\"q_index\\\":
1, \\\"timestamp\\\": \\\"2025-08-09T20:44:30.187799Z\\\"}\\n{\\\"_meta\\\":
{\\\"session_id\\\": \\\"sess-20231011162620\\\", \\\"ts\\\": \\\"2025-08-
09T20:45:24.222404Z\\\", \\\"n_records\\\": 2}}\\n{\\\"speaker\\\": \\\"interviewer\\\",
\\\"message\\\": \\\"QUESTION: Are there any additional external tools or APIs that
may be optional but beneficial for the system?\\\", \\\"subject\\\": \\\"Integrations
& External Systems\\\", \\\"q_index\\\": 2, \\\"timestamp\\\": \\\"2025-08-
09T20:45:24.222384Z\\\"}\\n{\\\"speaker\\\": \\\"user\\\", \\\"message\\\": \\\"whatsapp
support or any other social networks, but this is out of scope for this
release\\\", \\\"subject\\\": \\\"Integrations & External Systems\\\", \\\"q_index\\\":
2, \\\"timestamp\\\": \\\"2025-08-09T20:45:24.222384Z\\\"}\\n{\\\"_meta\\\":
{\\\"session_id\\\": \\\"sess-20231011162620\\\", \\\"ts\\\": \\\"2025-08-
09T20:46:32.224100Z\\\", \\\"n_records\\\": 2}}\\n{\\\"speaker\\\": \\\"interviewer\\\",
\\\"message\\\": \\\"QUESTION: Are there any legal, regulatory compliance, or
environmental constraints shaping this system's assumptions or
functionality?\\\", \\\"subject\\\": \\\"Constraints & Assumptions\\\", \\\"q_index\\\":
1, \\\"timestamp\\\": \\\"2025-08-09T20:46:32.224083Z\\\"}\\n{\\\"speaker\\\":
\\\"user\\\", \\\"message\\\": \\\"yesit has to be complaint with GDPR, PIII and all
data has to be encrypted also calls will be done from 8am to 5pm only, all offers
will be predefined and available in a database, this offer can be updated by the
marketting team as desired, There wil be prededined responses in the database and
also additional information of preview experiences in the RAG that the AI dialler
could use\\\", \\\"subject\\\": \\\"Constraints & Assumptions\\\", \\\"q_index\\\": 1,
\\\"timestamp\\\": \\\"2025-08-09T20:46:32.224083Z\\\"}\\n{\\\"_meta\\\":
{\\\"session_id\\\": \\\"sess-20231011162620\\\", \\\"ts\\\": \\\"2025-08-
09T20:47:32.000309Z\\\", \\\"n_records\\\": 2}}\\n{\\\"speaker\\\": \\\"interviewer\\\",
\\\"message\\\": \\\"QUESTION: Are there any assumptions about resource
availability, stakeholder responsibilities, or system limitations we should
document?\\\", \\\"subject\\\": \\\"Constraints & Assumptions\\\", \\\"q_index\\\": 2,
\\\"timestamp\\\": \\\"2025-08-09T20:47:32.000288Z\\\"}\\n{\\\"speaker\\\": \\\"user\\\",
\\\"message\\\": \\\"no\\\", \\\"subject\\\": \\\"Constraints & Assumptions\\\",
\\\"q_index\\\": 2, \\\"timestamp\\\": \\\"2025-08-
09T20:47:32.000288Z\\\"}\\n{\\\"_meta\\\": {\\\"session_id\\\": \\\"sess-
20231011162620\\\", \\\"ts\\\": \\\"2025-08-09T20:49:41.536324Z\\\", \\\"n_records\\\":
2}}\\n{\\\"speaker\\\": \\\"interviewer\\\", \\\"message\\\": \\\"QUESTION: What are the
potential risks or dependencies we should be aware of for this project?\\\",
\\\"subject\\\": \\\"Dependencies & Risks\\\", \\\"q_index\\\": 1, \\\"timestamp\\\":
\\\"2025-08-09T20:49:41.536303Z\\\"}\\n{\\\"speaker\\\": \\\"user\\\", \\\"message\\\":
\\\"we depnd on cloud providers to host the dialer and SIP trunk providers for the
calls\\\", \\\"subject\\\": \\\"Dependencies & Risks\\\", \\\"q_index\\\": 1,
\\\"timestamp\\\": \\\"2025-08-09T20:49:41.536303Z\\\"}\\n{\\\"_meta\\\":
{\\\"session_id\\\": \\\"sess-20231011162620\\\", \\\"ts\\\": \\\"2025-08-
09T20:51:30.864574Z\\\", \\\"n_records\\\": 2}}\\n{\\\"speaker\\\": \\\"interviewer\\\",
\\\"message\\\": \\\"QUESTION: How will these dependencies on cloud and SIP trunk

```

```

providers be managed to avoid downtime or disruptions?\\", \\\"subject\\\":
\\\"Dependencies & Risks\\\", \\\"q_index\\\": 2, \\\"timestamp\\\": \\\"2025-08-
09T20:51:30.864558Z\\\"}\\n{\\\"speaker\\\": \\\"user\\\", \\\"message\\\": \\\"yes we
have two different provider for cloud and two for SIP trunks\\\", \\\"subject\\\":
\\\"Dependencies & Risks\\\", \\\"q_index\\\": 2, \\\"timestamp\\\": \\\"2025-08-
09T20:51:30.864558Z\\\"}\\n\", \"annotations\": null}}]', name='read_file',
call_id='call_crcgWvONfwnhyp66ILOj8F2v', is_error=False)]
----- ToolCallSummaryMessage (terminate_agent) -----
[{"type": "text", "text": "{\\\"_meta\\\": {\\\"session_id\\\": \\\"sess-20231011162620\\\",
\\\"ts\\\": \\\"2025-08-09T20:35:33.157973Z\\\", \\\"n_records\\\": 2}}\\n{\\\"speaker\\\":
\\\"interviewer\\\", \\\"message\\\": \\\"QUESTION: Can you briefly describe the new
product and its primary purpose?\\\", \\\"subject\\\": \\\"Product Overview\\\",
\\\"q_index\\\": 1, \\\"timestamp\\\": \\\"2025-08-09T20:35:33.157947Z\\\"}\\n{\\\"speaker\\\":
\\\"user\\\", \\\"message\\\": \\\"The new product is an AI dialer that calls customers to
offer travel packages. If the customer accepts, the call transfers to a human. If
declined, the AI attempts to convince the customer up to three additional times
before ending the call\\\", \\\"subject\\\": \\\"Product Overview\\\", \\\"q_index\\\": 1,
\\\"timestamp\\\": \\\"2025-08-09T20:35:33.157947Z\\\"}\\n{\\\"_meta\\\": {\\\"session_id\\\":
\\\"sess-20231011162620\\\", \\\"ts\\\": \\\"2025-08-09T20:36:12.337556Z\\\", \\\"n_records\\\":
2}}\\n{\\\"speaker\\\": \\\"interviewer\\\", \\\"message\\\": \\\"QUESTION: What are the key
features or functionalities that you envision for this AI dialer?\\\", \\\"subject\\\":
\\\"Product Overview\\\", \\\"q_index\\\": 2, \\\"timestamp\\\": \\\"2025-08-
09T20:36:12.337533Z\\\"}\\n{\\\"speaker\\\": \\\"user\\\", \\\"message\\\": \\\"scope in: the
dialer will be integrated with the call center system to make and transfer calls,
with the CRM to get customer information like name, phone number, location,
salary, with a database to get the pre defined offers and to save calls resulst
as succesful call, closed deals, rejected deals, call errors and a RAG to answer
customer doubts and questions, t the end of the shift the AI shouuld make a
report with metrics and KPIs, scope out: whats app integration, social network
integration\\\", \\\"subject\\\": \\\"Product Overview\\\", \\\"q_index\\\": 2, \\\"timestamp\\\":
\\\"2025-08-09T20:36:12.337533Z\\\"}\\n{\\\"_meta\\\": {\\\"session_id\\\": \\\"sess-
20231011162620\\\", \\\"ts\\\": \\\"2025-08-09T20:36:40.339321Z\\\", \\\"n_records\\\":
2}}\\n{\\\"speaker\\\": \\\"interviewer\\\", \\\"message\\\": \\\"QUESTION: How would you define
success for this project in terms of measurable business objectives?\\\",
\\\"subject\\\": \\\"KPI & Success Metrics\\\", \\\"q_index\\\": 1, \\\"timestamp\\\": \\\"2025-08-
09T20:36:40.339292Z\\\"}\\n{\\\"speaker\\\": \\\"user\\\", \\\"message\\\": \\\"the KPI will be
configurable and metrics will inlcude succesfull calls, error calls, succesfull
deal, rejected deals, the reprot should include chats, tables and a dynamic
dashboard will all KPI and metrics, Performance: the syetem should be able to
support up to 300 call simultanously, Reliability:have an avaiabiity of 99.96%,
responses have to be in real time, and use natural language and human like voice
quality, Usability: use best prctices, Security:use access control, dev full
access to app and code, marketting to reports\\\", \\\"subject\\\": \\\"KPI & Success
Metrics\\\", \\\"q_index\\\": 1, \\\"timestamp\\\": \\\"2025-08-
09T20:36:40.339292Z\\\"}\\n{\\\"_meta\\\": {\\\"session_id\\\": \\\"sess-20231011162620\\\",

```

```

\"ts\": \"2025-08-09T20:37:13.210804Z\", \"n_records\": 2}}\n{\"speaker\": 
\"interviewer\", \"message\": \"QUESTION: Should additional reporting 
capabilities or real-time data visualization options be part of the MVP?\", 
\"subject\": \"KPI & Success Metrics\", \"q_index\": 2, \"timestamp\": \"2025-08-
09T20:37:13.210784Z\"}\n{\"speaker\": \"user\", \"message\": \"no\", \"subject\": 
\"KPI & Success Metrics\", \"q_index\": 2, \"timestamp\": \"2025-08-
09T20:37:13.210784Z\"}\n{\"_meta\": {\"session_id\": \"sess-20231011162620\", 
\"ts\": \"2025-08-09T20:37:44.574598Z\", \"n_records\": 2}}\n{\"speaker\": 
\"interviewer\", \"message\": \"QUESTION: Are there any functionalities or 
integrations that can be delayed for future versions?\", \"subject\": \"Scope In 
and Out\", \"q_index\": 1, \"timestamp\": \"2025-08-
09T20:37:44.574581Z\"}\n{\"speaker\": \"user\", \"message\": \"scope in: the 
dialer will be integrated with the call center system to make and transfer calls, 
with the CRM to get customer information like name, phone number, location, 
salary, with a database to get the pre defined offers and to save calls resulst 
as succesful call, closed deals, rejected deals, call errors and a RAG to answer 
customer doubts and questions, t the end of the shift the AI should make a 
report with metrics and KPIs\", \"subject\": \"Scope In and Out\", \"q_index\": 
1, \"timestamp\": \"2025-08-09T20:37:44.574581Z\"}\n{\"_meta\": {\"session_id\": 
\"sess-20231011162620\", \"ts\": \"2025-08-09T20:38:20.921468Z\", \"n_records\": 
2}}\n{\"speaker\": \"interviewer\", \"message\": \"QUESTION: Are there specific 
use cases or customer demographics the product should prioritize first?\", 
\"subject\": \"Scope In and Out\", \"q_index\": 2, \"timestamp\": \"2025-08-
09T20:38:20.921445Z\"}\n{\"speaker\": \"user\", \"message\": \"That will be 
configurable and will depend on each campaign\", \"subject\": \"Scope In and 
Out\", \"q_index\": 2, \"timestamp\": \"2025-08-
09T20:38:20.921445Z\"}\n{\"_meta\": {\"session_id\": \"sess-20231011162620\", 
\"ts\": \"2025-08-09T20:38:55.328313Z\", \"n_records\": 2}}\n{\"speaker\": 
\"interviewer\", \"message\": \"QUESTION: Are there any specific regulatory 
requirements or compliance standards the system must adhere to?\", \"subject\": 
\"Non-Functional Requirements\", \"q_index\": 1, \"timestamp\": \"2025-08-
09T20:38:55.328292Z\"}\n{\"speaker\": \"user\", \"message\": \"Performance: the 
syetem should be able to support up to 300 call simultanously, Reliability:have 
an avaiabiity of 99.96%, responses have to be in real time, and use natural 
language and human like voice quality, Usability: use best prctices, Security:use 
access control, dev full access to app and code, marketting to reports\", 
\"subject\": \"Non-Functional Requirements\", \"q_index\": 1, \"timestamp\": 
\"2025-08-09T20:38:55.328292Z\"}\n{\"_meta\": {\"session_id\": \"sess-
20231011162620\", \"ts\": \"2025-08-09T20:41:11.244248Z\", \"n_records\": 
2}}\n{\"speaker\": \"interviewer\", \"message\": \"QUESTION: Are there additional 
constraints like language support, accessibility, or anything else we should 
consider?\", \"subject\": \"Non-Functional Requirements\", \"q_index\": 2, 
\"timestamp\": \"2025-08-09T20:41:11.244225Z\"}\n{\"speaker\": \"user\", 
\"message\": \"it will be only english, voice quality has to be human like level, 
the tone has to be respectful but convingin\", \"subject\": \"Non-Functional

```

```

Requirements\", \"q_index\": 2, \"timestamp\": \"2025-08-
09T20:41:11.244225Z\"}\n{\"_meta\": {\"session_id\": \"sess-20231011162620\",
\"ts\": \"2025-08-09T20:41:32.082905Z\", \"n_records\": 2}}\n{\"speaker\":
\"interviewer\", \"message\": \"QUESTION: Can you describe the main user roles
and their interactions with the AI dialer?\", \"subject\": \"User Roles &
Personas\", \"q_index\": 1, \"timestamp\": \"2025-08-
09T20:41:32.082886Z\"}\n{\"speaker\": \"user\", \"message\": \"the users wil be
Travel agencies and airlines, the project stake holders are, marketting team
provide pre defined offers and pre defined answers and documentation and past
surveys for RAG, marketting VP is the sponsor, It team for support and
development, human agents to take transfer and close deals are testers, and the
final custoemr from the companies that will receive the call\", \"subject\":
\"User Roles & Personas\", \"q_index\": 1, \"timestamp\": \"2025-08-
09T20:41:32.082886Z\"}\n{\"_meta\": {\"session_id\": \"sess-20231011162620\",
\"ts\": \"2025-08-09T20:43:11.256210Z\", \"n_records\": 2}}\n{\"speaker\":
\"interviewer\", \"message\": \"QUESTION: Are there specific user personas or
scenarios that need to be addressed in the system design?\", \"subject\": \"User
Roles & Personas\", \"q_index\": 2, \"timestamp\": \"2025-08-
09T20:43:11.256187Z\"}\n{\"speaker\": \"user\", \"message\": \"what I already
told\", \"subject\": \"User Roles & Personas\", \"q_index\": 2, \"timestamp\":
\"2025-08-09T20:43:11.256187Z\"}\n{\"_meta\": {\"session_id\": \"sess-
20231011162620\", \"ts\": \"2025-08-09T20:44:30.187823Z\", \"n_records\":
2}}\n{\"speaker\": \"interviewer\", \"message\": \"QUESTION: What systems or
external integrations are necessary for the AI dialer to function effectively?\",
\"subject\": \"Integrations & External Systems\", \"q_index\": 1, \"timestamp\":
\"2025-08-09T20:44:30.187799Z\"}\n{\"speaker\": \"user\", \"message\": \"the
dialer will be integrated with the call center system to make and transfer calls,
with the CRM to get customer information like name, phone number, location,
salary, with a database to get the pre defined offers and to save calls
resulst\", \"subject\": \"Integrations & External Systems\", \"q_index\": 1,
\"timestamp\": \"2025-08-09T20:44:30.187799Z\"}\n{\"_meta\": {\"session_id\":
\"sess-20231011162620\", \"ts\": \"2025-08-09T20:45:24.222404Z\", \"n_records\":
2}}\n{\"speaker\": \"interviewer\", \"message\": \"QUESTION: Are there any
additional external tools or APIs that may be optional but beneficial for the
system?\", \"subject\": \"Integrations & External Systems\", \"q_index\": 2,
\"timestamp\": \"2025-08-09T20:45:24.222384Z\"}\n{\"speaker\": \"user\",
\"message\": \"whatsapp support or any other social networks, but this is out of
scope for this release\", \"subject\": \"Integrations & External Systems\",
\"q_index\": 2, \"timestamp\": \"2025-08-09T20:45:24.222384Z\"}\n{\"_meta\":
{\"session_id\": \"sess-20231011162620\", \"ts\": \"2025-08-
09T20:46:32.224100Z\", \"n_records\": 2}}\n{\"speaker\": \"interviewer\",
\"message\": \"QUESTION: Are there any legal, regulatory compliance, or
environmental constraints shaping this system's assumptions or functionality?\",
\"subject\": \"Constraints & Assumptions\", \"q_index\": 1, \"timestamp\":
\"2025-08-09T20:46:32.224083Z\"}\n{\"speaker\": \"user\", \"message\": \"yesit

```

```

has to be complaint with GDPR, PIII and all data has to be encrypted also calls
will be done from 8am to 5pm only, all offers will be predefined and available in
a database, this offer can be updated by the marketting team as desired, There
will be prededined responses in the database and also additional information of
preview experiences in the RAG that the AI dialler could use\", \"subject\":
\"Constraints & Assumptions\", \"q_index\": 1, \"timestamp\": \"2025-08-
09T20:46:32.224083Z\"}\\n{\\\"_meta\\\": {\\\"session_id\\\": \\\"sess-20231011162620\\\",
\\\"ts\\\": \\\"2025-08-09T20:47:32.000309Z\\\", \\\"n_records\\\": 2}}\\n{\\\"speaker\\\":
\\\"interviewer\\\", \\\"message\\\": \\\"QUESTION: Are there any assumptions about
resource availability, stakeholder responsibilities, or system limitations we
should document?\\\", \\\"subject\\\": \\\"Constraints & Assumptions\\\", \\\"q_index\\\": 2,
\\\"timestamp\\\": \\\"2025-08-09T20:47:32.000288Z\\\"}\\n{\\\"speaker\\\": \\\"user\\\",
\\\"message\\\": \\\"no\\\", \\\"subject\\\": \\\"Constraints & Assumptions\\\", \\\"q_index\\\": 2,
\\\"timestamp\\\": \\\"2025-08-09T20:47:32.000288Z\\\"}\\n{\\\"_meta\\\": {\\\"session_id\\\":
\\\"sess-20231011162620\\\", \\\"ts\\\": \\\"2025-08-09T20:49:41.536324Z\\\", \\\"n_records\\\":
2}}\\n{\\\"speaker\\\": \\\"interviewer\\\", \\\"message\\\": \\\"QUESTION: What are the
potential risks or dependencies we should be aware of for this project?\\\",
\\\"subject\\\": \\\"Dependencies & Risks\\\", \\\"q_index\\\": 1, \\\"timestamp\\\": \\\"2025-08-
09T20:49:41.536303Z\\\"}\\n{\\\"speaker\\\": \\\"user\\\", \\\"message\\\": \\\"we depnd on cloud
providers to host the dialer and SIP trunk providers for the calls\\\",
\\\"subject\\\": \\\"Dependencies & Risks\\\", \\\"q_index\\\": 1, \\\"timestamp\\\": \\\"2025-08-
09T20:49:41.536303Z\\\"}\\n{\\\"_meta\\\": {\\\"session_id\\\": \\\"sess-20231011162620\\\",
\\\"ts\\\": \\\"2025-08-09T20:51:30.864574Z\\\", \\\"n_records\\\": 2}}\\n{\\\"speaker\\\":
\\\"interviewer\\\", \\\"message\\\": \\\"QUESTION: How will these dependencies on cloud
and SIP trunk providers be managed to avoid downtime or disruptions?\\\",
\\\"subject\\\": \\\"Dependencies & Risks\\\", \\\"q_index\\\": 2, \\\"timestamp\\\": \\\"2025-08-
09T20:51:30.864558Z\\\"}\\n{\\\"speaker\\\": \\\"user\\\", \\\"message\\\": \\\"yes we have two
different provider for cloud and two for SIp trunks\\\", \\\"subject\\\":
\\\"Dependencies & Risks\\\", \\\"q_index\\\": 2, \\\"timestamp\\\": \\\"2025-08-
09T20:51:30.864558Z\\\"}\\n\", \"annotations\": null}]
----- ToolCallRequestEvent (terminate_agent) -----
[FunctionCall(id='call_G4MHfvho5e03jVWxCoTgfv6Q',
arguments='{\"path\":\"/run/media/jun/Jun_6T/Code/TFM_BA_GPT/experience/sess-
20231011162620\"}', name='create_directory')]
----- ToolCallExecutionEvent (terminate_agent) -----
[FunctionExecutionResult(content='[{\"type\": \"text\", \"text\": \"Successfully created
directory /run/media/jun/Jun_6T/Code/TFM_BA_GPT/experience/sess-20231011162620\",
\"annotations\": null}]', name='create_directory',
call_id='call_G4MHfvho5e03jVWxCoTgfv6Q', is_error=False)]
----- ToolCallSummaryMessage (terminate_agent) -----
[{\"type\": \"text\", \"text\": \"Successfully created directory
/run/media/jun/Jun_6T/Code/TFM_BA_GPT/experience/sess-20231011162620\",
\"annotations\": null}]
----- ToolCallRequestEvent (terminate_agent) -----

```

```
[FunctionCall(id='call_8aud6cDcveFNoeUs5kE2PIA7', arguments={'source':
"/run/media/jun/Jun_6T/Code/TFM_BA_GPT/conversation/interview.jsonl",
"destination": "/run/media/jun/Jun_6T/Code/TFM_BA_GPT/experience/sess-
20231011162620/interview.jsonl"}', name='move_file'),
FunctionCall(id='call_aW5aFc3T5AGyFw4Bhrrm4m6w', arguments={'source':
"/run/media/jun/Jun_6T/Code/TFM_BA_GPT/conversation/functional_specification.md",
"destination": "/run/media/jun/Jun_6T/Code/TFM_BA_GPT/experience/sess-
20231011162620/functional_specification.md"}', name='move_file'),
FunctionCall(id='call_npz1flH409j9AztZ04IdD6xD', arguments={'source':
"/run/media/jun/Jun_6T/Code/TFM_BA_GPT/conversation/process_diagram.mmd",
"destination": "/run/media/jun/Jun_6T/Code/TFM_BA_GPT/experience/sess-
20231011162620/process_diagram.mmd"}', name='move_file')]
----- ToolCallExecutionEvent (terminate_agent) -----
[FunctionExecutionResult(content='[{"type": "text", "text": "Successfully moved
/run/media/jun/Jun_6T/Code/TFM_BA_GPT/conversation/interview.jsonl to
/run/media/jun/Jun_6T/Code/TFM_BA_GPT/experience/sess-
20231011162620/interview.jsonl", "annotations": null}]', name='move_file',
call_id='call_8aud6cDcveFNoeUs5kE2PIA7', is_error=False),
FunctionExecutionResult(content='[{"type": "text", "text": "Successfully moved
/run/media/jun/Jun_6T/Code/TFM_BA_GPT/conversation/functional_specification.md to
/run/media/jun/Jun_6T/Code/TFM_BA_GPT/experience/sess-
20231011162620/functional_specification.md", "annotations": null}]',
name='move_file', call_id='call_aW5aFc3T5AGyFw4Bhrrm4m6w', is_error=False),
FunctionExecutionResult(content='[{"type": "text", "text": "Successfully moved
/run/media/jun/Jun_6T/Code/TFM_BA_GPT/conversation/process_diagram.mmd to
/run/media/jun/Jun_6T/Code/TFM_BA_GPT/experience/sess-
20231011162620/process_diagram.mmd", "annotations": null}]', name='move_file',
call_id='call_npz1flH409j9AztZ04IdD6xD', is_error=False)]
----- ToolCallSummaryMessage (terminate_agent) -----
[{"type": "text", "text": "Successfully moved
/run/media/jun/Jun_6T/Code/TFM_BA_GPT/conversation/interview.jsonl to
/run/media/jun/Jun_6T/Code/TFM_BA_GPT/experience/sess-
20231011162620/interview.jsonl", "annotations": null}]
[{"type": "text", "text": "Successfully moved
/run/media/jun/Jun_6T/Code/TFM_BA_GPT/conversation/functional_specification.md to
/run/media/jun/Jun_6T/Code/TFM_BA_GPT/experience/sess-
20231011162620/functional_specification.md", "annotations": null}]
[{"type": "text", "text": "Successfully moved
/run/media/jun/Jun_6T/Code/TFM_BA_GPT/conversation/process_diagram.mmd to
/run/media/jun/Jun_6T/Code/TFM_BA_GPT/experience/sess-
20231011162620/process_diagram.mmd", "annotations": null}]
----- ToolCallRequestEvent (terminate_agent) -----
[FunctionCall(id='call_t3Epp1q5i9844uqNQ5BbTSh5',
arguments={'input_path': "/run/media/jun/Jun_6T/Code/TFM_BA_GPT/experience/sess-
20231011162620/process_diagram.mmd", "output_path": "/run/media/jun/Jun_6T/Code/TFM
```

```

_BA_GPT/experience/sess-20231011162620/process_diagram.mmd", "overwrite": true}',
name='clean_mermaid'])
----- ToolCallExecutionEvent (terminate_agent) -----
[FunctionExecutionResult(content='[{"type": "text", "text":
"{\\"written_to\\":\\"/run/media/jun/Jun_6T/Code/TFM_BA_GPT/experience/sess-
20231011162620/process_diagram.mmd\\",\\"bytes\\":942,\\"detected_type\\":\\"flow
chart\\",\\"issues\\":[{\\"severity\\":\\"info\\",\\"code\\":\\"DEDENTED\\",\\"me
ssage\\":\\"Common left indent
removed.\\",\\"recommendation\\":\\"None\\",\\"line\\":1}]}", "annotations":
null}]', name='clean_mermaid', call_id='call_t3Epp1q5i9844uqNQ5BbTSh5',
is_error=False)]
----- ToolCallSummaryMessage (terminate_agent) -----
[{"type": "text", "text":
"{\\"written_to\\":\\"/run/media/jun/Jun_6T/Code/TFM_BA_GPT/experience/sess-
20231011162620/process_diagram.mmd\\",\\"bytes\\":942,\\"detected_type\\":\\"flowchart\\
",\\"issues\\":[{\\"severity\\":\\"info\\",\\"code\\":\\"DEDENTED\\",\\"message\\":\\"Common
left indent removed.\\",\\"recommendation\\":\\"None\\",\\"line\\":1}]}", "annotations":
null}]
----- ToolCallRequestEvent (terminate_agent) -----
[FunctionCall(id='call_4RYE5fEdr6OY1ulsAqRkYxN6',
arguments={'input_path':"/run/media/jun/Jun_6T/Code/TFM_BA_GPT/experience/sess-
20231011162620/process_diagram.mmd", "output_path":"/run/media/jun/Jun_6T/Code/TFM
_BA_GPT/experience/sess-20231011162620/process_diagram.png", "theme":"forest"}',
name='render_mermaid_png')]
----- ToolCallExecutionEvent (terminate_agent) -----
[FunctionExecutionResult(content='[{"type": "text", "text": "PNG written to
/run/media/jun/Jun_6T/Code/TFM_BA_GPT/experience/sess-
20231011162620/process_diagram.png", "annotations": null}]',
name='render_mermaid_png', call_id='call_4RYE5fEdr6OY1ulsAqRkYxN6',
is_error=False)]
----- ToolCallSummaryMessage (terminate_agent) -----
[{"type": "text", "text": "PNG written to
/run/media/jun/Jun_6T/Code/TFM_BA_GPT/experience/sess-
20231011162620/process_diagram.png", "annotations": null}]
----- ToolCallRequestEvent (terminate_agent) -----
[FunctionCall(id='call_dQPSmYN1mPOxo3Paunl4lkYa',
arguments={'session_dir':"/run/media/jun/Jun_6T/Code/TFM_BA_GPT/experience/sess-
20231011162620", "md":"functional_specification.md", "diagram":"process_diagram.png
", "md_out":"functional_specification.formatted.md", "pdf_out":"deliverable.pdf", "e
ngine":"xelatex", "toc":true, "toc_depth":2, "margin":"1in", "mainfont":"DejaVu
Serif", "monofont":"DejaVu Sans Mono"}', name='format_and_export')]
----- ToolCallExecutionEvent (terminate_agent) -----
[FunctionExecutionResult(content='[{"type": "text", "text":
"{\\"written_to\\":\\"/run/media/jun/Jun_6T/Code/TFM_BA_GPT/experience/sess-
20231011162620/functional_specification.formatted.md\\",\\"business_requirements\

```

```

\":5,\\\"functional_requirements\\\":7,\\\"diagram\\\":\\\"process_diagram.png\\\",\\\"p
df\\\":{\\\"written_to\\\":\\\"/run/media/jun/Jun_6T/Code/TFM_BA_GPT/experience/sess-
20231011162620/deliverable.pdf\\\",\\\"stdout\\\":\\\"\\\"}}\", \"annotations\": null}}',
name='format_and_export', call_id='call_dQPSmYN1mPOxo3Paunl4lkYa',
is_error=False)]
----- ToolCallSummaryMessage (terminate_agent) -----
[{"type": "text", "text":
"{\\\"written_to\\\":\\\"/run/media/jun/Jun_6T/Code/TFM_BA_GPT/experience/sess-
20231011162620/functional_specification.formatted.md\\\",\\\"business_requirements\\\":
5,\\\"functional_requirements\\\":7,\\\"diagram\\\":\\\"process_diagram.png\\\",\\\"pdf\\\":{\\\"wr
itten_to\\\":\\\"/run/media/jun/Jun_6T/Code/TFM_BA_GPT/experience/sess-
20231011162620/deliverable.pdf\\\",\\\"stdout\\\":\\\"\\\"}}\", \"annotations\": null}}]
----- ToolCallRequestEvent (terminate_agent) -----
[FunctionCall(id='call_IszrQEvWuvZCNTIoL5KG4n2P',
arguments='{\"path\":\"/run/media/jun/Jun_6T/Code/TFM_BA_GPT/experience/sess-
20231011162620/deliverable.pdf\"}', name='get_file_info')]
----- ToolCallExecutionEvent (terminate_agent) -----
[FunctionExecutionResult(content='[{"type": "text", "text": "size:
89854\\ncreated: 2025-09-06 21:38:16.761930\\nmodified: 2025-09-06
21:38:16.761930\\naccessed: 2025-09-06 21:23:36.030949\\nisDirectory:
False\\nisFile: True\\npermissions: 644", "annotations": null}]',
name='get_file_info', call_id='call_IszrQEvWuvZCNTIoL5KG4n2P', is_error=False)]
----- ToolCallSummaryMessage (terminate_agent) -----
[{"type": "text", "text": "size: 89854\\ncreated: 2025-09-06
21:38:16.761930\\nmodified: 2025-09-06 21:38:16.761930\\naccessed: 2025-09-06
21:23:36.030949\\nisDirectory: False\\nisFile: True\\npermissions: 644",
"annotations": null}]
----- ToolCallRequestEvent (terminate_agent) -----
[FunctionCall(id='call_Vo3UIHSan9apsFqFXsIKK7Q',
arguments='{\"session_dir\":\"/run/media/jun/Jun_6T/Code/TFM_BA_GPT/experience/sess-
20231011162620\", \"session_id\":\"sess-20231011162620\"}',
name='save_experience_chroma')]
----- ToolCallExecutionEvent (terminate_agent) -----
[FunctionExecutionResult(content='[{"type": "text", "text":
"{\\\"ok\\\":true,\\\"session_id\\\":\\\"sess-
20231011162620\\\",\\\"files_indexed\\\":6,\\\"written_items\\\":22,\\\"collection\\\":\\\"
\\\"tfm_ba_gpt_experiences\\\",\\\"persistence_path\\\":\\\"/run/media/jun/Jun_6T/Code/
TFM_BA_GPT/.experiences\\\"}", "annotations": null}]',
name='save_experience_chroma', call_id='call_Vo3UIHSan9apsFqFXsIKK7Q',
is_error=False)]
----- ToolCallSummaryMessage (terminate_agent) -----
[{"type": "text", "text": "{\\\"ok\\\":true,\\\"session_id\\\":\\\"sess-
20231011162620\\\",\\\"files_indexed\\\":6,\\\"written_items\\\":22,\\\"collection\\\":\\\"tfm_ba
_gpt_experiences\\\",\\\"persistence_path\\\":\\\"/run/media/jun/Jun_6T/Code/TFM_BA_GPT/.
experiences\\\"\", \"annotations\": null}]

```

```
----- TextMessage (terminate_agent) -----  
**TERMINATE**
```

Contents

Functional Specification Document	2
Title Page	2
Introduction	3
Scope	4
In-Scope Features	4
Business Requirements	4
Out-of-Scope Features	4
Functional Requirements	5
Functional Requirements Mapping	6
Non-Functional Requirements	6
User Roles & Personas	7
Internal Users	7
External Users	7
Assumptions & Constraints	8
Assumptions	8
Constraints	8
Use Cases	9
Use Case 1: Make an Outbound Call	9
Use Case 2: Generate Call Reports	10
Glossary	11
Business Terms	11
Technical Terms	11
System Features	11
General Terms	11
Process Diagram	12

Functional Specification Document

Title Page

Project Name: AI Dialer for Travel Packages

Version: 1.0

Date: 2023-11-03

Introduction

The purpose of this project is to develop an AI-powered dialer system designed to call customers and offer travel packages. The AI will handle call interactions, transfer customers to human agents for finalizing deals, and aim to improve efficiency and customer engagement. By leveraging integrations with existing systems, the AI dialer will streamline outbound campaigns and provide comprehensive reporting on KPIs.

Scope

In-Scope Features

1. Integration with the call center to:
 - Initiate and transfer calls.
 - Support up to 300 simultaneous calls.
2. Integration with CRM systems for accessing the following customer details:
 - Name, phone number, location, and salary.
3. Database usage for:
 - Predefined offers and responses.
 - Storing call results (successful, rejected, or error calls).
 - Storing additional resources for generating responses to customer questions using a Rules and Guidance System (RAG).
4. Generation of end-of-shift metrics and KPIs, which include:
 - Successful calls, errors, successful deals, and rejected deals.
 - Reports featuring charts, tables, and a dynamic dashboard.
5. Real-time decision-making based on predefined offers and call status.

Business Requirements

Business Requirement ID	Description
BR1	Integration with the call center to: Initiate and transfer calls.; Support up to 300 simultaneous calls.
BR2	Integration with CRM systems for accessing the following customer details: Name, phone number, location, and salary.
BR3	Database usage for: Predefined offers and responses.; Storing call results (successful, rejected, or error calls).; Storing additional resources for generating responses to customer questions using a Rules and Guidance System (RAG).
BR4	Generation of end-of-shift metrics and KPIs, which include: Successful calls, errors, successful deals, and rejected deals.; Reports featuring charts, tables, and a dynamic dashboard.
BR5	Real-time decision-making based on predefined offers and call status.

Out-of-Scope Features

1. Integration with WhatsApp or other social networks.
2. Expansion to non-English languages.

Functional Requirements

1. The AI dialer must integrate with the call center system to initiate, manage, and transfer calls in real time.
2. The AI must access relevant customer data from the CRM system during the call execution process.
3. The system must store and retrieve predefined offers and responses from a database.
4. The AI should generate reports at the end of each shift, summarizing KPIs and metrics.
5. The system should allow marketing teams to update offers and predefined responses in the database.
6. Calls should operate only within the hours of 8:00 AM to 5:00 PM.
7. All data, including call results and customer interactions, must be stored securely with encryption.

Functional Requirements Mapping

Business Requirement ID	Functional Requirement ID	Description
BR1	FR1	The AI dialer must integrate with the call center system to initiate, manage, and transfer calls in real time.
BR2	FR2	The AI must access relevant customer data from the CRM system during the call execution process.
BR3	FR3	The system must store and retrieve predefined offers and responses from a database.
BR4	FR4	The AI should generate reports at the end of each shift, summarizing KPIs and metrics.
BR5	FR5	The system should allow marketing teams to update offers and predefined responses in the database.
	FR6	Calls should operate only within the hours of 8:00 AM to 5:00 PM.
	FR7	All data, including call results and customer interactions, must be stored securely with encryption. —

Non-Functional Requirements

1. **Performance:**
 - Support up to 300 simultaneous calls.
2. **Availability and Reliability:**
 - Ensure a system availability of 99.96%.
3. **Language & Voice Quality:**
 - Support only the English language.
 - Use natural language processing to achieve human-like voice quality with respectful and convincing tones.
4. **Security:**
 - Comply with GDPR and PII regulations.
 - Implement access controls: developers have full access, while the marketing team has restricted access to reports.
5. **Usability:**
 - Follow best practices for UI/UX design.

User Roles & Personas

Internal Users

1. **Marketing Team:**
 - Updates predefined offers, answers, and reports.
2. **Human Agents:**
 - Handle transferred calls and close deals.
3. **IT Team:**
 - Provide technical support and ongoing system maintenance.
4. **Sponsor:**
 - Marketing VP oversees funding and priority alignment.

External Users

1. **Travel Agencies and Airlines:**
 - Organizations using the system to connect with end customers.
2. **Final Customers:**
 - Recipients of the AI dialer calls.

Assumptions & Constraints

Assumptions

1. The marketing team will provide predefined offers and customer surveys to input in the database.
2. The IT team has suitable resources for system deployment and maintenance.

Constraints

1. Calls will be restricted to working hours (8:00 AM-5:00 PM).
2. The system must comply with GDPR and PII regulations.
3. All offers and answers are predefined and stored in the database.

Use Cases

Use Case 1: Make an Outbound Call

Description: The AI Dialer enables human agents to make outbound calls efficiently to customers, providing behavioral insights to tailor the call experience.

Actors:

- Human Agents
- Customers

Preconditions:

1. The agent must be logged into the AI Dialer system.
2. The customer's contact information must exist in the database.
3. The outbound call feature must be enabled for the agent.

Postconditions:

1. The customer receives the call.
2. The system records call details such as duration, outcome, and notes.

Main Flow:

1. The agent initiates the outbound call through the AI Dialer system.
2. The AI Dialer retrieves the customer's contact information from the database.
3. The system analyzes the mood and behavior of the customer using past interactions or real-time data.
4. The dialer connects the call to the customer.
5. The agent communicates with the customer.
6. At the end of the call, the system prompts the agent to record notes and the call outcome.
7. If applicable, the system suggests follow-up actions based on the call's result.

Alternate Flows:

1. **Call Fails to Connect:** 1.1 The system retries the call three times before marking it as "Failed to Connect." 1.2 A notification is sent to the agent regarding failure.
2. **Customer Busy:** 2.1 The system schedules a callback and notifies the agent.

Use Case 2: Generate Call Reports

Description: The AI Dialer provides detailed call reports for the marketing team and IT support to analyze performance and troubleshoot issues.

Actors:

- Marketing Teams
- IT Support

Preconditions:

1. Reports must have data from completed calls.
2. The user must have access permissions to view reports.

Postconditions:

1. A summarized report is generated with metrics like call duration, success rate, and agent performance.
2. Users can download or share the report for further analysis.

Main Flow:

1. The user logs into the AI Dialer system.
2. The user navigates to the reporting module.
3. The user requests a custom report by selecting filters such as date range, agent ID, or call outcomes.
4. The system processes and generates the report.
5. The user views the report on the screen and downloads it if necessary.

Alternate Flows:

1. **Missing Data:** 1.1 The system alerts the user about incomplete or missing records relevant to the report. 1.2 The user can adjust the filters or wait for the data to refresh.
2. **Export Error:** 2.1 The system retries the export process automatically. 2.2 If unsuccessful, the system prompts the user to make manual corrections.

Glossary

Business Terms

- **CRM (Customer Relationship Management):** A system or software that manages a company's interactions with current and potential customers. In the context of AI Dialer, CRM integration allows for optimized customer outreach and data handling.
- **KPI (Key Performance Indicator):** Metrics used to evaluate the success of specific activities or overall performance. In the AI Dialer system, KPIs could include call success rates, conversion rates, and average call handling time.
- **GDPR Compliance (General Data Protection Regulation Compliance):** Adherence to data protection and privacy laws established by the European Union. For AI Dialers, this means safeguarding customer information during data storage, transmission, and processing.

Technical Terms

- **SIP (Session Initiation Protocol):** A communication protocol used for initiating, maintaining, and terminating real-time sessions, like voice and video calls. Crucial for AI Dialers in interacting with VoIP systems.
- **Natural Language Processing (NLP):** A field of artificial intelligence focused on enabling computers to understand, interpret, and respond to human language. In AI Dialers, NLP is used for interpreting customer speech and generating appropriate responses.
- **RAG (Red-Amber-Green):** A status indicator framework frequently used in project management. For AI Dialers, it could represent live monitoring of system performance or issue resolution status.

System Features

- **Predefined Responses:** A library of canned or templated answers programmed into the AI Dialer for common queries or circumstances, ensuring quick and consistent communication.
- **Navigation:** Tools and features within the AI Dialer interface that assist users in moving through the system efficiently.

General Terms

- **Communication Protocols:** Standards governing the exchange of information between systems and devices. In AI Dialer software, robust protocols ensure reliable and flexible communication processes.
- **Interoperability:** The capability of the AI Dialer system to integrate with other software or systems (e.g., CRM, VoIP platforms) seamlessly.

Process Diagram

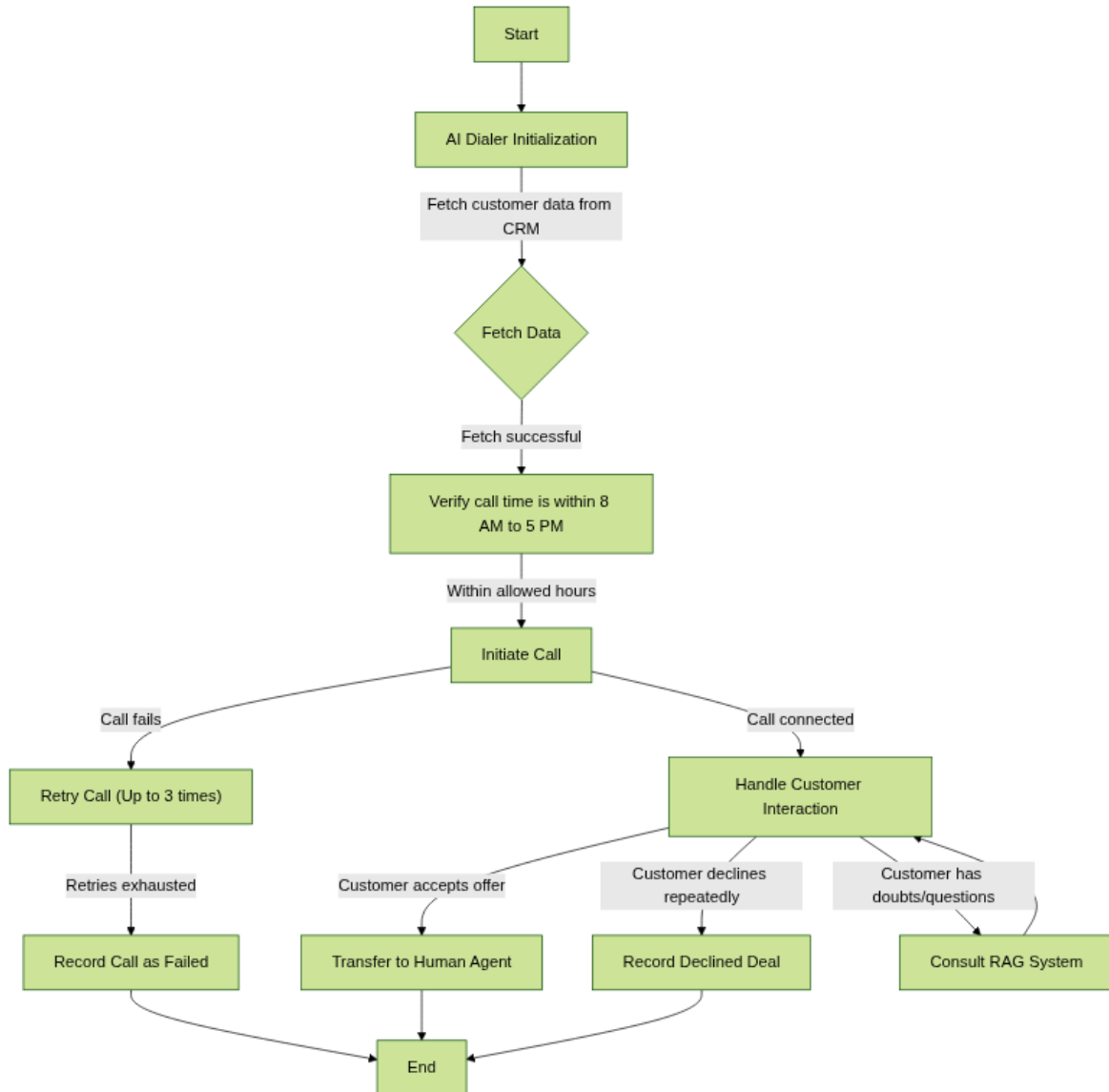


Figure 1: Process Diagram