

Universidad de Castilla-La Mancha

Escuela Superior de Ingeniería Informática de Albacete

Escuela Superior de Informática de Ciudad Real

Trabajo Fin de Máster

Máster Universitario en Inteligencia Artificial

Diseño y Desarrollo de un Sistema Experimental de Automatización de Estrategias de Inversión Sobre la API de Interactive Brokers

David Cebrián Sierra

Octubre 2026

Escuela Superior de Ingeniería Informática Albacete

Escuela Superior de Informática

TRABAJO FIN DE MÁSTER

Máster Universitario en Inteligencia Artificial

Diseño y Desarrollo de un Sistema Experimental de Automatización de Estrategias de Inversión Sobre la API de Interactive Brokers

Autor: David Cebrián Sierra

Director: Santiago Sánchez Sobrino

Codirector: David Vallejo Fernández

UCLM Universidad Castilla-La Mancha

Octubre 2026

A todo el que me apoya
me aguanta y cree en mi.
Te quiero baby.

Declaración de autoría

Yo, David Cebrián Sierra, con DNI 47070175P, declaro que soy el único autor del trabajo fin de master titulado Diseño y Desarrollo de un Sistema Experimental de Automatización de Estrategias de Inversión Sobre la API de Interactive Brokers, que el citado trabajo no infringe las leyes en vigor sobre propiedad intelectual, y que todo el material no original contenido en dicho trabajo está apropiadamente atribuido a sus legítimos autores.

Albacete, a 1 de Octubre de 2026

Fdo.: David Cebrián Sierra

Resumen

Este Trabajo Fin de Máster presenta el diseño, implementación y validación de un roboadvisor experimental que explota la API de Interactive Brokers y que está basado en una arquitectura multi-agente de inversión orientado a los mercados IBEX35 y S&P500. El sistema integra seis agentes buscadores sustentados en modelos de lenguaje de gran escala (LLM) con soporte tanto para inferencia local como vía API, un motor de evaluación multi-perfil, un pipeline de decisión determinístico y un mecanismo de supervisión humana en el bucle (HITL) mediante un bot de Telegram. La ejecución de órdenes se realiza sobre Interactive Brokers en modo *paper trading* a través de la librería `ib_async`.

El núcleo del sistema sigue el principio de *no-operar por defecto*: únicamente se emite una orden cuando la esperanza matemática calculada es positiva, todos los controles de riesgo cuantitativos están validados y el operador humano aprueba explícitamente la operación. La arquitectura separa de forma estricta el razonamiento heurístico (responsabilidad de los LLM) de la ejecución transaccional (código determinístico con validación previa de precios), garantizando la trazabilidad y el cumplimiento normativo exigido por MiFID II, el EU AI Act y el reglamento DORA.

La orquestación del flujo se implementa de manera dual mediante un grafo dirigido con puntos de suspensión en LangGraph y un orquestador determinístico equivalente en Python puro. La persistencia y el desacoplamiento se apoya en PostgreSQL/TimescaleDB, mientras que la observabilidad en tiempo real se cubre mediante once métricas Prometheus visualizadas en Grafana. Sobre esta infraestructura, un asistente de cartera basado en LLM genera informes periódicos por Telegram reconciliando el beneficio y pérdida realizado (P&L), atiende consultas en lenguaje natural y propone ajustes de configuración sujetos a confirmación interactiva.

Los resultados de la validación en *paper trading* demuestran la viabilidad del enfoque para detectar anomalías de mercado de forma autónoma, filtrarlas bajo estrictas barreras

cuantitativas y someterlas a deliberación humana con suficiente contexto cualitativo. El trabajo concluye con la identificación de líneas futuras de evolución, destacando la simulación avanzada de políticas globales de cartera (rebalanceo periódico, DCA dinámico y *stress tests*), la optimización adaptativa de umbrales mediante aprendizaje por refuerzo y la consolidación de la memoria semántica con *embeddings* persistentes.

Palabras clave: robo-advisor, sistema multi-agente, LLM local, inversión algorítmica, LangGraph, Interactive Brokers, Human-in-the-Loop, observabilidad, gobierno IA, Python.

Agradecimientos

La realización de este Trabajo Fin de Máster marca la culminación de una etapa de intenso aprendizaje, dedicación y crecimiento profesional y personal. Este proyecto no habría sido posible sin el apoyo, la orientación y la confianza de las personas que me han acompañado a lo largo de este camino.

En primer lugar, quiero expresar mi más sincero agradecimiento a mis directores de trabajo, por su guía experta, su rigor metodológico y su constante predisposición para orientar este proyecto hacia los más altos estándares académicos y técnicos.

A los profesores del Máster, por compartir su conocimiento y experiencia, despertando mi interés por áreas tan apasionantes y dinámicas como la inteligencia artificial, el aprendizaje automático y los sistemas distribuidos.

A mi pareja, Alba, por su apoyo incondicional, su paciencia infinita y su comprensión en cada una de las largas jornadas dedicadas a este proyecto. Gracias por estar siempre a mi lado, por ser mi mayor refugio y por creer en mí incluso en los momentos de mayor exigencia. Este logro también es tuyo.

A mi familia, por su aliento constante en cada paso, por celebrar cada avance y por ser siempre un pilar fundamental en todos mis proyectos y decisiones.

A mis amigos y compañeros de trabajo, por las conversaciones enriquecedoras, los ánimos en los momentos de mayor complejidad técnica y por recordar siempre el valor de mantener la curiosidad y la perspectiva.

A todos ellos, mi más profunda gratitud.

Lista de acrónimos

API	Application Programming Interface
HFT	High-Frequency Trading
HITL	Human-In-The-Loop
IBKR	Interactive Brokers
LLM	Large Language Model
OHLCV	Open, High, Low, Close, Volume
P&L	Profit and Loss
RL	Reinforcement Learning
SL	Stop-Loss
TFM	Trabajo Fin de Máster
TP	Take-Profit
TTL	Time-To-Live
DCA	Dollar-Cost Averaging
FPGA	Field-Programmable Gate Array
GRU	Gated Recurrent Unit
LSTM	Long Short-Term Memory
MCP	Model Context Protocol

Índice general

Lista de acrónimos . . . . . **xi**

1 Introducción **1**

1.1 Contexto y motivación 1

1.2 Planteamiento del problema y brecha tecnológica 2

1.3 Solución propuesta y principios de diseño 3

1.4 Procedimiento de realización y validación experimental 4

1.5 Impacto e implicaciones socioeconómicas 5

1.6 Estructura de la memoria 5

2 Estado del arte **7**

2.1 Evolución del trading algorítmico y horizontes operativos 7

2.2 Aprendizaje automático y procesamiento del lenguaje natural en finanzas. . . 8

2.2.1 De los modelos cuantitativos tradicionales a los modelos de series temporales . *8*

2.2.2 Procesamiento del lenguaje natural y análisis de sentimiento financiero *9*

2.3 Grandes modelos de lenguaje y sistemas multi-agente en finanzas 10

2.3.1 Modelos de lenguaje como motores de razonamiento *10*

2.3.2 Sistemas multi-agente (MAS) y marcos de orquestación *11*

2.4 Análisis crítico y evaluación del marco TradingAgents 11

2.4.1 Justificación metodológica del rechazo de TradingAgents como núcleo del sistema *12*

2.5 Posicionamiento comparativo y aportación diferencial 13

3 Objetivos y requisitos del sistema **15**

3.1 Objetivo general 15

3.2 Objetivos específicos 15
3.3 Requisitos funcionales 17
3.4 Requisitos no funcionales 17
3.5 Principios de diseño 20

4 Metodología 21

4.1 Enfoque de desarrollo 21
4.2 Diseño orientado a la testabilidad 21
4.3 Gestión de la configuración 22
4.4 Estrategia de validación 22
4.5 Justificación de las decisiones tecnológicas principales 23
4.5.1 Análisis comparativo y selección de la plataforma de intermediación 23
4.5.2 Python como lenguaje principal 25
4.5.3 DeepSeek v4-flash como LLM 26
4.5.4 LangGraph para la orquestación 26
4.5.5 ib_async para la conexión con IBKR 26
4.5.6 PostgreSQL/TimescaleDB para la persistencia 26

5 Arquitectura y diseño del sistema 27

5.1 Visión general 27
5.2 Capa de ingesta: buscadores especializados y control de flujo 28
5.3 Componentes del pipeline de decisión y filtros de riesgo 30
5.4 Supervisión interactiva (HITL) y ciclo de vida de la orden 32
5.5 Persistencia, observabilidad y asistente de cartera 33
5.6 Detalles relevantes de la implementación y diseño de software 34
5.6.1 Estructura modular del proyecto 34
5.6.2 Diseño guiado por contratos de datos inmutables (Pydantic v2) 34
5.6.3 Inyección de dependencias y pruebas offline 35
5.6.4 Dualidad en la orquestación y suspensión nativa en LangGraph 35
5.6.5 Resiliencia ante el Error 162 de Interactive Brokers 35
5.6.6 Reconciliación contable FIFO de ejecuciones reales 36

6 Evaluación y resultados 37

6.1 Verificación funcional y batería de pruebas automatizadas 37
6.2 Evaluación cuantitativa: motor de backtesting y simulación 39
6.3 Operación experimental en paper trading y análisis del embudo de decisión . 40

6.4 Análisis de costes del proyecto 43
6.4.1 Costes de personal (Ingeniería de software) *43*
6.4.2 Costes de hardware, licencias e infraestructura *43*
6.4.3 Presupuesto global *44*

7 Conclusiones y trabajo futuro 45

7.1 Cumplimiento de objetivos y competencias del Máster 45
7.2 Alineamiento con las competencias del Máster 47
7.3 Conclusiones principales y lecciones aprendidas 47
7.4 Contribuciones del trabajo 48
7.5 Líneas de trabajo futuro 49
7.5.1 Optimización adaptativa, memoria semántica y aprendizaje continuo *49*
7.5.2 Políticas globales de gestión de cartera y simulación de escenarios *49*

A Configuración del Sistema 51

A.1 Fichero principal: config.yaml 51
A.2 Fichero de límites de riesgo: risk_limits.yaml 54
A.3 Fichero de pesos del decisor: decisor_weights.yaml 55
A.4 Variables de entorno (.env) 55

B Métricas y Observabilidad 57

B.1 Métricas Prometheus completas 57
B.2 Dashboard de Grafana 58
B.3 Dashboard Streamlit: agent_monitor.py 59
B.4 Audit Logger 59

Referencia bibliográfica 63

Índice de figuras

5.1 Diagrama de contenedores del sistema. 28
5.2 Máquina de estados determinística del pipeline de decisión. 31
5.3 Diagrama de secuencia del ciclo de decisión y autorización interactiva HITL. 33

6.1 Embudo de decisión: conversión progresiva de señales hacia órdenes de mercado. 41
6.2 Recreación de la interfaz del bot de Telegram en el momento de solicitar aprobación humana (HITL). 42

Índice de tablas

2.1 Comparativa del sistema desarrollado frente a enfoques del estado del arte 14

3.1 Requisitos funcionales del sistema 18
3.2 Requisitos no funcionales del sistema 19

4.1 Análisis preliminar de plataformas de inversión y conectividad programática 24
4.2 Comparativa técnica de candidatos viables para agentes de inversión asistidos por LLM 25

5.1 Especificaciones operativas de los seis buscadores de mercado 29
5.2 Ponderaciones y umbrales del motor de decisión (`decisor_weights.yaml`) 32

6.1 Cobertura y escenarios de la batería de pruebas automatizadas 38
6.2 Resumen del presupuesto global de desarrollo e implantación del sistema 44

B.1 Métricas Prometheus del sistema (detalle completo) 58

1. Introducción

1.1. Contexto y motivación

La gestión cuantitativa y la operativa en mercados financieros han experimentado una transformación sin precedentes durante la última década [LdP18, BS24]. Históricamente, las estrategias sistemáticas de inversión requerían una infraestructura computacional masiva, conectividad de ultra-baja latencia co-localizada en los parqués y equipos multidisciplinares de matemáticos y desarrolladores, restringiendo su despliegue casi con exclusividad a grandes fondos de cobertura y mesas de tesorería institucionales [Ald13].

Sin embargo, la convergencia reciente de tres factores tecnológicos ha alterado este equilibrio de poder:

1. **Apertura y democratización de interfaces de negociación**: la consolidación de APIs estables en brókeres minoristas e institucionales (como Interactive Brokers), junto con la disponibilidad de flujos de datos de mercado en tiempo real y datos fundamentales a coste accesible.
2. **Madurez de los modelos de lenguaje de gran escala (Large Language Model (LLM)) como motores de inferencia contextual**: la capacidad de modelos avanzados para procesar lenguaje natural heterogéneo (noticias de última hora, comunicados macroeconómicos, sentimiento en redes sociales y ratios corporativos), traduciendo información no estructurada en representaciones cuantitativas accionables [W+23, YLW23].
3. **Ecosistemas de orquestación multi-agente**: la aparición de patrones y marcos de trabajo basados en grafos dirigidos (como LangGraph), que permiten articular arquitecturas modulares donde múltiples entidades especializadas colaboran, mantienen memoria de contexto y gestionan estados persistentes [Lan24].

Este nuevo paradigma abre la puerta al diseño de una nueva generación de **robo-advisors experimentales**: sistemas autónomos capaces no solo de ejecutar rebalanceos mecánicos de carteras pasivas, sino de analizar el contexto global del mercado, formular hipótesis tácticas de inversión intradía y argumentar sus conclusiones en lenguaje natural ante el inversor.

No obstante, esta creciente autonomía operativa introduce riesgos operacionales y sistémicos de primer orden [KKST17]. La historia del trading automatizado está jalonada de episodios críticos —desde el célebre *Flash Crash* de mayo de 2010 hasta incidentes causados por agentes de aprendizaje por refuerzo (Reinforcement Learning (RL)) enfrentados a cambios imprevistos de régimen de volatilidad [Int24]—. Cuando se delega la toma de decisiones en modelos de caja negra o agentes probabilísticos que pueden sufrir alucinaciones o derivas de razonamiento, la ausencia de barreras de contención determinísticas puede acarrear la liquidación fulminante del capital del operador.

Por consiguiente, la automatización en los mercados financieros no puede concebirse como un fin en sí mismo, sino que debe subordinarse inexcusablemente a estrictos principios de gobernanza, supervisión humana en el bucle (Human-In-The-Loop (HITL)) y control de riesgo determinístico [DSC25].

1.2. Planteamiento del problema y brecha tecnológica

El auge de los agentes basados en LLM ha propiciado propuestas académicas recientes como FinAgent [Z+24], TradingGPT [L+23] o TradingAgents [XL25]. Si bien evidencian el potencial de los modelos de lenguaje para sintetizar información, en la actualidad no existe una solución efectiva al problema de la inversión automatizada con agentes LLM en entornos reales.

La literatura existente se encuentra atrapada en una dicotomía insalvable que nadie ha resuelto de forma conjunta:

- Por un lado, los sistemas algorítmicos cuantitativos tradicionales poseen una estricta rigidez matemática y ejecutan en mercado con solvencia, pero son por completo ciegos al contexto cualitativo (incapaces de interpretar una noticia geopolítica, un rumor en redes o una comparecencia de un banco central).
- Por otro lado, los marcos multi-agente de LLM actuales (como TradingAgents) operan como meros prototipos conversacionales sobre datos estáticos: confían la gestión del riesgo a «debates dialécticos» entre agentes sintéticos en lenguaje natural, carecen de filtros matemáticos inviolables, no gestionan universos amplios de activos

en tiempo real ni disponen de una capa de ejecución resiliente conectada a brókeres reales, incumpliendo además las exigencias regulatorias europeas de auditoría e interrupción inmediata (*kill switch*) impuestas por MiFID II, el EU AI Act y DORA [Par14, Eur24, Eur22].

A partir de esta brecha no resuelta, el problema central que aborda este Trabajo Fin de Máster se formula en los siguientes términos:

> **Problema central:** *¿Es posible diseñar e implementar un robo-advisor experimental que articule una arquitectura multi-agente basada en LLM, que explote el API de Interactive Brokers, para la detección y evaluación de oportunidades de inversión sobre universos amplios de mercado (IBEX35 y S&P500), garantizando que la ejecución de órdenes quede estrictamente blindada por filtros de riesgo matemáticos determinísticos, supervisión humana interactiva y cumplimiento por diseño de las directrices de MiFID II, EU AI Act y DORA?*

1.3. Solución propuesta y principios de diseño

Para resolver esta brecha, la nueva perspectiva adoptada en este trabajo consiste en concebir un paradigma híbrido con desacoplamiento estricto: los modelos generativos se restringen a labores de exploración heurística y razonamiento cualitativo, mientras que la admisión de riesgo, la ponderación del consenso y el envío de órdenes quedan supeditados exclusivamente a un núcleo de código determinístico y a la aprobación humana.

La arquitectura se fundamenta en cinco principios directores:

1. **Principio de no-operar por defecto**: en mercados financieros, la inacción preserva el capital. Ante cualquier incertidumbre, fallo de conectividad, dato incompleto o discrepancia evaluadora, el estado es invariablemente `NO_OPERATE`. Solo se procesa una propuesta cuando todas las señales convergen positivamente, la esperanza matemática proyectada es favorable, se superan los filtros cuantitativos y el operador autoriza la orden.
2. **Desacoplamiento estricto entre razonamiento y ejecución**: los agentes basados en LLM proponen ideas tras procesar fuentes heterogéneas (noticias, redes, técnicos y fundamentos). Por el contrario, la agregación, el filtro de riesgo (VaR, correlación, liquidez), el cálculo del consenso y el ruteo de órdenes corresponden a módulos deterministicos sin variabilidad probabilística. La inteligencia generativa propone; el código determinístico dispone.

3. **Human-in-the-Loop (HITL) interactivo y asíncrono**: se articula un canal mediante un bot de Telegram que notifica al operador las oportunidades consensuadas con niveles de entrada, Take-Profit (TP), Stop-Loss (SL), ratio riesgo/beneficio (R/R) y justificación textual. La orden queda suspendida en el grafo hasta que el operador pulsa un botón interactivo, aplicando una caducidad temporal (Time-To-Live (TTL) de 300 segundos) que descarta la orden si no hay confirmación humana explícita.
4. **Observabilidad, auditoría y gobierno continuo**: registro inmutable de cada ciclo en base de datos de series temporales (TimescaleDB) mediante un bus de eventos (Redis Streams), junto con la exportación de once métricas Prometheus visualizadas en Grafana.
5. **Asistente conversacional de cartera y banco de simulación**: un agente asesor basado en LLM reconcilia el Profit and Loss (P&L) liquidado directamente desde los *fills* del bróker bajo contabilidad FIFO, emite informes periódicos en lenguaje natural y permite interactuar mediante consultas libres y barridos de simulación sobre cotizaciones históricas agregadas.

1.4. Procedimiento de realización y validación experimental

La propuesta se materializa a través del desarrollo de un nuevo artefacto software modular de código abierto implementado en Python 3.13. Su contraste experimental y validación empírica se articula en tres etapas:

1. **Verificación funcional y suite de pruebas offline**: construcción de una batería de 21 ficheros de tests unitarios e integración que, mediante inyección de dependencias y clientes simulados (*mocks*), valida el comportamiento determinístico de los filtros de riesgo y la suspensión del grafo sin requerir conectividad de red ni costes de inferencia.
2. **Banco de simulación cuantitativa**: modelización del ciclo de vida de órdenes bracket sobre 252 sesiones bursátiles sintéticas y barras de mercado reales agregadas desde el bróker, cuantificando métricas estandarizadas de rendimiento financiero (Sharpe, drawdown máximo, Calmar y esperanza matemática).
3. **Validación experimental en *paper trading***: despliegue del sistema completo conectado en tiempo real a la infraestructura de Interactive Brokers (IB Gateway) monitorizando 508 activos (IBEX35 y S&P500), verificando la resiliencia operativa ante contingencias de red (como el Error 162) y analizando el embudo de conversión de señales bajo supervisión humana efectiva.

1.5. Impacto e implicaciones socioeconómicas

El desarrollo de este trabajo trasciende la mera aportación de ingeniería de software al incidir en tres dimensiones de relevancia socioeconómica:

- **Democratización del acceso a herramientas de análisis institucional**: históricamente, la capacidad de procesar flujos masivos de información multimodal en tiempo real ha estado restringida a grandes firmas financieras mediante infraestructuras privativas de costes millonarios. Este proyecto demuestra que es viable articular herramientas de vigilancia y análisis contextual avanzado para inversores individuales o gestoras medianas utilizando tecnologías de código abierto e interfaces estándar accesibles.
- **Protección del inversor minorista y mitigación de asimetrías**: los operadores particulares son especialmente vulnerables al sesgo emocional (FOMO, pánico en correcciones) y a la manipulación en redes sociales. El principio de «no-operar por defecto», junto con la exigencia de justificación argumentada y el cálculo estricto de la esperanza matemática, proporciona una barrera disciplinada que antepone sistemáticamente la preservación del capital a la especulación impulsiva.
- **Gobernanza y adopción responsable de la IA financiera**: en un contexto donde los organismos reguladores internacionales advierten sobre los riesgos sistémicos de la automatización descontrolada y las alucinaciones de los modelos generativos, este trabajo aporta un modelo de referencia de cómo conciliar la adopción de la inteligencia artificial con el cumplimiento estricto del marco normativo europeo (MiFID II, EU AI Act y DORA). Demuestra que la supervisión humana (Human-in-the-Loop) y los cortacircuitos determinísticos no son un freno a la innovación, sino el requisito fundacional para una integración segura y ética de la IA en la economía productiva.

1.6. Estructura de la memoria

El resto de esta memoria se organiza de la siguiente forma:

- **Capítulo 2 (Estado del arte)**: revisión del estado del arte en trading algorítmico, sistemas multi-agente, modelos de lenguaje y marco regulatorio (MiFID II, EU AI Act, DORA).
- **Capítulo 3 (Objetivos y requisitos)**: definición formal de los objetivos del proyecto y requisitos funcionales y no funcionales del sistema.

- **Capítulo 4 (Metodología)**: metodología de desarrollo iterativa orientada a contratos, entorno de trabajo, comparativa de brókeres y justificación tecnológica.
- **Capítulo 5 (Arquitectura y diseño del software)**: diseño modular por capas, flujo del pipeline de decisión, salvaguardas de riesgo determinísticas, nodo HITL y decisiones de ingeniería de software.
- **Capítulo 6 (Evaluación y resultados)**: proceso de validación con tests automatizados, resultados del motor de simulación y sesiones de *paper trading*.
- **Capítulo 7 (Conclusiones y trabajo futuro)**: balance del cumplimiento de objetivos y competencias, síntesis de contribuciones, reflexión personal y líneas maestras de trabajo futuro.
- **Anexos A y B**: configuración del sistema y métricas Prometheus.

2. Estado del arte

Este capítulo presenta una revisión crítica de la literatura científica, los desarrollos tecnológicos y los marcos normativos que sustentan la investigación en sistemas automatizados de inversión. Se estructura en seis ejes principales: la evolución histórica de la negociación algorítmica y sus horizontes operativos; la transición de modelos cuantitativos clásicos hacia el procesamiento del lenguaje natural financiero; la emergencia de agentes basados en modelos de lenguaje de gran escala (LLM); el análisis en profundidad del marco de referencia reciente *TradingAgents*; y, finalmente, el posicionamiento diferencial del presente trabajo.

2.1. Evolución del trading algorítmico y horizontes operativos

El concepto de trading algorítmico —definido como la toma o ejecución automatizada de decisiones de inversión mediante reglas codificadas en software— experimentó su génesis en la década de 1970 con la introducción de sistemas automáticos de casación de órdenes y la implantación pionera del sistema *Designated Order Turnaround* (DOT) en la Bolsa de Nueva York en 1976 [HJM11]. Durante las décadas posteriores, la progresiva desregulación financiera en los mercados occidentales y el abaratamiento computacional propiciaron el nacimiento del arbitraje estadístico y de los primeros sistemas de seguimiento de tendencias puramente sistemáticos.

Hacia finales de la década de 2000, la convergencia entre infraestructuras de telecomunicaciones avanzadas y el acceso directo a los libros de órdenes culminó en la consolidación del High-Frequency Trading (HFT) [Ald13]. En este régimen operativo, las decisiones se toman en escalas de microsegundos o nanosegundos mediante algoritmos ejecutados en hardware especializado (e.g., matrices Field-Programmable Gate Array (FPGA)) y servidores físicamente co-localizados en los centros de datos de los mercados rectores. Si bien el HFT

absorbió una atención mediática y regulatoria sustancial, transformó la ventaja competitiva en una carrera puramente tecnológica de reducción de latencia de red, dejando al margen a operadores minoristas y gestoras medianas debido a las barreras económicas insalvables de infraestructura.

En contraposición, el segmento de negociación de baja frecuencia (*low-frequency trading*) y horizonte intradía o swing (de horas a varios días) basa su viabilidad en la calidad de la inferencia sobre el contexto de mercado más que en la velocidad de acceso al libro de órdenes [CC24, Aru23]. En este régimen, los operadores explotan ineficiencias causadas por asimetrías de información, sobre-reacciones psicológicas a eventos informativos y dinámicas de momentum inter-mercado. La literatura actual subraya que es en este horizonte operativo donde las técnicas modernas de aprendizaje automático y procesamiento del lenguaje natural ofrecen su mayor potencial transformador, al permitir sintetizar masivamente flujos de información cualitativa que un analista humano no puede procesar en tiempo real.

2.2. Aprendizaje automático y procesamiento del lenguaje natural en finanzas

2.2.1. De los modelos cuantitativos tradicionales a los modelos de series temporales

La aplicación de algoritmos de aprendizaje supervisado a la predicción de series temporales financieras ha sido ampliamente documentada en las últimas dos décadas. No obstante, los mercados financieros presentan desafíos singulares: una baja relación señal-ruido, propiedades de no-estacionariedad estricta y cambios recurrentes de régimen de mercado.

López de Prado [LdP18] ha sistematizado de forma crítica las razones por las cuales una gran proporción de los modelos cuantitativos publicados en la literatura fracasan estrepitosamente en entornos de negociación en vivo. Entre los factores determinantes destacan:

- El sobreajuste masivo provocado por la optimización de hiperparámetros sobre series históricas no estacionarias sin purgado de datos.
- La fuga de información futura (data leakage) al aplicar técnicas estándar de validación cruzada (k-fold) que destruyen la correlación temporal intrínseca de los retornos.
- La ausencia de filtros de liquidez y de modelado de costes de deslizamiento y comisiones de intermediación.

Históricamente, los primeros intentos por superar los modelos lineales autorregresivos

(ARIMA, GARCH) recurrieron a redes neuronales recurrentes, particularmente las variantes Long Short-Term Memory (LSTM) y Gated Recurrent Unit (GRU) [FK18]. Aunque estas arquitecturas capturan dependencias temporales complejas a corto y medio plazo, adolecen de una fragilidad predictiva acusada fuera de la distribución de entrenamiento, siendo fácilmente desestabilizadas por choques de volatilidad imprevistos.

En paralelo, los algoritmos basados en árboles de decisión potenciados por gradiente (Gradient Boosted Decision Trees, como XGBoost y LightGBM) se han consolidado como herramientas de alto rendimiento en predicción tabular financiera [CG16], ofreciendo mayor robustez frente al sobreajuste en presencia de datos ruidosos y conjuntos de muestra limitados.

Recientemente, el paradigma de los modelos fundacionales preentrenados sobre series temporales —destacando Chronos [A$^+$24] y TimeGPT— ha introducido capacidades de predicción zero-shot y few-shot. A pesar de sus prometedoras propiedades de generalización en pronósticos de demanda o telemetría, su adaptación a la generación de señales bursátiles accionables continúa siendo objeto de debate metodológico, debido a la dificultad inherente de anticipar perturbaciones exógenas causadas por eventos informativos discretos.

2.2.2. Procesamiento del lenguaje natural y análisis de sentimiento financiero

Dado que una fracción sustancial de los movimientos en los precios de los activos se desencadena por información textual no estructurada (noticias corporativas, comparecencias de bancos centrales o informes trimestrales), el Procesamiento del Lenguaje Natural se ha convertido en un componente insustituible del análisis cuantitativo contemporáneo.

El trabajo seminal de Tetlock [Tet07] sentó las bases empíricas del área al evidenciar cuantitativamente que un elevado volumen de palabras de sesgo pesimista en columnas financieras predice descensos significativos en los índices bursátiles durante la jornada subsiguiente, seguido de reversiones a la media. Los primeros sistemas comerciales dependían de diccionarios léxicos predefinidos (como el léxico financiero de Loughran y McDonald). Sin embargo, estos enfoques carecían de comprensión contextual, siendo incapaces de interpretar el tono condicional, la negación o la ambigüedad semántica propia de la jerga económica.

El advenimiento de la arquitectura Transformer supuso una revolución cualitativa. Araci [Ara19] introdujo FinBERT, un modelo derivado de BERT preentrenado y ajustado específicamente sobre corpus textuales financieros (Reuters, declaraciones SEC), logrando una precisión marcadamente superior en la clasificación de sentimiento y polaridad financiera en

comparación con cualquier método basado en reglas o modelos genéricos.

En años más recientes, la escalabilidad de los modelos fundacionales ha dado lugar a arquitecturas a gran escala de dominio específico, tales como BloombergGPT [W+23] —un modelo de 50 mil millones de parámetros entrenado sobre un corpus híbrido financiero y de dominio general— y FinGPT [YLW23], una iniciativa orientada a democratizar los modelos financieros mediante estrategias de ajuste por instrucciones (instruction fine-tuning) de código abierto. Estos modelos no se limitan a asignar etiquetas de polaridad positiva o negativa, sino que proporcionan explicaciones analíticas, identifican entidades nombradas y extraen relaciones complejas entre variables macroeconómicas.

2.3. Grandes modelos de lenguaje y sistemas multi-agente en finanzas

2.3.1. Modelos de lenguaje como motores de razonamiento

El salto cualitativo definitivo en el uso de los LLM ha sido su transición desde simples clasificadores textuales hacia auténticos motores de razonamiento y planificación. Gracias al aprendizaje en contexto (in-context learning), técnicas como el razonamiento por pasos (Chain-of-Thought, CoT) y la alternancia iterativa entre razonamiento y ejecución de herramientas (patrón ReAct) [Y+23] permiten que un modelo descomponga problemas financieros complejos en subtareas verificables.

En este marco, trabajos recientes han explorado la aplicación de agentes basados en LLM a la toma de decisiones de inversión:

- FinAgent [Z+24]: propone un agente multimodal con módulos avanzados de memoria episódica y herramientas externas para la operativa en mercados de criptoactivos, integrando análisis visual de gráficos con noticias.
- TradingGPT [L+23]: evalúa el comportamiento de modelos como GPT-4 en la generación continua de carteras a partir del análisis en cascada de titulares y datos macroeconómicos.

A pesar de sus logros conceptuales, estos desarrollos adolecen frecuentemente de una deficiencia crítica: confunden la capacidad de generar hipótesis persuasivas con la capacidad de tomar decisiones operativas seguras. Un LLM aislado es propenso a sufrir alucinaciones fácticas, deriva en los cálculos aritméticos (e.g., errores en el cálculo de distancias de stop-loss o dimensionamiento de lotes) y carece de un modelo interno de correlación matemática

entre activos.

2.3.2. Sistemas multi-agente (MAS) y marcos de orquestación

La teoría de los Sistemas Multi-Agente (*Multi-Agent Systems*, MAS) cuenta con un dilatado recorrido en la economía computacional, donde históricamente se han utilizado modelos basados en agentes (*Agent-Based Modeling*, ABM) para simular dinámicas de mercado emergentes, formación de burbujas especulativas y comportamiento de manada [LeB06].

La unión entre sistemas multi-agente y modelos de lenguaje permite distribuir la carga cognitiva: en lugar de exigir a un único modelo monolítico que asuma todos los roles del ciclo inversor, se instancian agentes especializados (analistas de noticias, técnicos, evaluadores de riesgo y gestores de ejecución) que colaboran o contrastan sus hipótesis.

No obstante, las primeras aproximaciones basadas en agentes autónomos sin restricciones —como AutoGPT [T$^+$23] o BabyAGI— evidenciaron problemas severos de divergencia, bucles infinitos y fallos impredecibles. Para aplicaciones de misión crítica como la gestión de capital, la comunidad ha virado hacia marcos de orquestación basados en grafos dirigidos, cuyo exponente principal es LangGraph [Lan24].

LangGraph redefine el ciclo del agente modelándolo como un grafo computacional con estados fuertemente tipados, ciclos condicionales controlados y, fundamentalmente, nodos de interrupción nativos (checkpoints y suspensión de estado). Esto permite entrelazar de forma natural procesos de razonamiento probabilístico (LLM) con barreras de control determinísticas y mecanismos de supervisión humana antes de transicionar hacia estados irreversibles (como el envío de órdenes de compra/venta).

2.4. Análisis crítico y evaluación del marco TradingAgents

Dentro de la literatura emergente en plataformas multi-agente, el proyecto TradingAgents v0.3.0 [XL25] constituye un referente de obligada revisión metodológica. TradingAgents implementa una arquitectura sobre LangGraph (`StateGraph`) donde un estado compartido (`AgentState`) es evaluado por doce agentes especializados organizados en cuatro escalones jerárquicos:

1. **Analyst Team**: agentes orientados a la recopilación e interpretación de fundamentos contables, noticias, redes sociales y señales técnicas.
2. **Research Team**: estructurado como un debate dialéctico adversarial donde un analista alcista (Bull Agent) y un analista bajista (Bear Agent) defienden sus tesis ante un

moderador (Research Manager).

3. **Trader Agent**: sintetiza el dictamen del debate y redacta una propuesta de transacción específica.
4. **Risk Team & Portfolio Management**: debate colegiado entre perfiles de riesgo diferenciados (agresivo, conservador y neutro) cuya sentencia final es dictada por un juez de riesgo (Risk Judge), emitiendo una recomendación de tipo `BUY`, `HOLD` o `SELL`.

2.4.1. Justificación metodológica del rechazo de TradingAgents como núcleo del sistema

Tras un examen detallado del código fuente y de la dinámica operativa de TradingAgents v0.3.0, en el presente Trabajo Fin de Máster (TFM) se determinó formalmente descartar dicho marco como base u orquestador central del proyecto, relegándolo a lo sumo a una función periférica de análisis documental puntual. Las razones arquitectónicas y operativas que fundamentan esta decisión son:

1. **Ausencia de escaneo continuo sobre universos reales de mercado**: TradingAgents está concebido como una herramienta de consulta bajo demanda orientada a analizar un único activo (*single ticker*) sobre una fecha concreta prefijada. No dispone de primitivas para monitorizar de forma asíncrona y simultánea cientos de activos (como la totalidad del IBEX35 o el S&P500), ni gestiona la cadencia periódica entre agentes buscadores.
2. **Inexistencia de un filtro de riesgo matemático determinístico**: En TradingAgents, el control de riesgo se delega en un debate dialéctico en lenguaje natural entre agentes LLM. Desde una perspectiva de ingeniería financiera y cumplimiento normativo, este planteamiento resulta inasumible: las restricciones de riesgo (apalancamiento, exposición máxima por activo, correlación matricial y Value-at-Risk) deben ser barreras matemáticas infranqueables calculadas en código determinístico, no argumentos retóricos sujetos a sesgos persuasivos o fallos de juicio generativo.
3. **Inviabilidad en términos de coste computacional y latencia operativa**: El flujo secuencial de debates conversacionales con múltiples turnos de palabra genera un volumen masivo de llamadas a las Application Programming Interfaces (APIs) de los modelos, con latencias acumuladas que superan frecuentemente los tres a cinco minutos por consulta y un consumo prohibitivo de tokens. Este comportamiento choca frontalmente con el requisito de tiempo de respuesta exigido para el trading intradía (< 60 segundos entre detección y tramitación).

4. **Carencia absoluta de capa de ejecución real y resiliencia**: TradingAgents se detiene en la emisión de un informe textual. Carece por completo de conectividad con plataformas de intermediación bursátil reales (Interactive Brokers), no modela tipos de órdenes vinculadas (órdenes bracket con Take-Profit y Stop-Loss obligatorios), no gestiona errores de producción del protocolo de mercado (como el Error 162 o límites de pacing) y omite los mecanismos de seguridad operacional (kill switch de emergencia y auditoría inmutable) exigidos por la legislación europea.

Esta divergencia sustancial impulsó el desarrollo de una arquitectura propia y original en este trabajo, orientada a la ingeniería de producción, la observabilidad continua y la contención matemática del riesgo.

2.5. Posicionamiento comparativo y aportación diferencial

Para sintetizar el estado del arte y resaltar la contribución específica de este proyecto frente a las alternativas existentes en la literatura y la industria, la tabla 2.1 recoge una comparativa multidimensional.

La aportación principal de este Trabajo Fin de Máster radica, por tanto, en cerrar la brecha existente entre la literatura académica sobre agentes de lenguaje y las exigencias de un sistema de producción auditable. Se demuestra la viabilidad de articular un sistema que aúna la riqueza heurística de los modelos de lenguaje con la seguridad inquebrantable del código determinístico, permitiendo la exploración de estrategias automatizadas de forma responsable y rigurosamente controlada.

Tabla 2.1: Comparativa del sistema desarrollado frente a enfoques del estado del arte

Característica	Robo-advisors clásicos	Sistemas ML cuantitativos	TradingAgents / FinAgent	Este trabajo
Naturaleza del motor	Reglas fijas y rebalanceo pasivo	Modelos estadísticos y redes neuronales	LLMs en debate conversacional	Multi-agente LLM + Núcleo determinístico
Fuentes analizadas	Cuestionario de perfilado	Series de precios y ratios tabulares	Noticias, finanzas y sentimiento	Noticias, redes, técnico, pre-market y fundamentos
Gestión del riesgo	Asignación estática de activos	Stop-loss estadístico / Volatilidad	Juicio conversacional de agentes LLM	Filtro matemático estricto (6 barreras + VaR)
Supervisión humana	Previa a la inversión inicial	Ninguna / Supervisión externa	Ninguna en el bucle operativo	HITL asíncrono interactivo con TTL vía Telegram
Conexión a bróker	Propietaria de la entidad	Conexión directa automática	Inexistente (simulación en texto)	Interactive Brokers en vivo (*paper trading*)
Cumplimiento normativo	MiFID II general	Parcial / Difícil auditabilidad	No contemplado	MiFID II (art. 17), EU AI Act y DORA por diseño

3. Objetivos y requisitos del sistema

3.1. Objetivo general

El objetivo general de este TFM es **diseñar, implementar y validar un sistema software que explote el API de la plataforma de Interactive Brokers usando modelos de inteligencia artificial** que, a partir de los datos provistos por la API de Interactive Brokers y fuentes heterogéneas, permita experimentar con estrategias automatizadas de inversión y análisis de carteras. El sistema no busca sustituir la operativa de la plataforma de intermediación, sino ofrecer un entorno integral de simulación, gestión de riesgo y análisis cuantitativo asistido por un modelo de lenguaje (LLM) como asistente conversacional en lenguaje natural.

3.2. Objetivos específicos

Para alcanzar el objetivo general, el trabajo se desglosa en los siguientes diez objetivos específicos, organizados según las fases del ciclo de ingeniería y alineados con la propuesta formativa del Máster:

OE1. Análisis de la API de Interactive Brokers y arquitectura de conectividad: analizar en profundidad las capacidades, protocolos y limitaciones de la API de Interactive Brokers (autenticación, consulta de saldos y carteras, tipología de órdenes y restricciones de *pacing*), implementando una capa de integración asíncrona mediante `ib_async` capaz de gestionar contingencias operativas (como el Error 162) y desacoplar las sesiones de consulta y ejecución.

OE2. Definición del modelo de datos y diseño por contratos: definir formalmente un modelo de datos fuertemente tipado mediante contratos inmutables (Pydantic v2) que

represente de forma unívoca carteras, posiciones, propuestas de inversión, evaluaciones de consenso, parámetros cuantitativos de riesgo y registros de ejecución (fills).

OE3. Desarrollo de la capa de ingesta y vigilancia de mercado: implementar seis agentes buscadores especializados capaces de escanear de forma asíncrona y periódica un universo de 508 activos (IBEX35 y S&P500), combinando indicadores cuantitativos y modelos de lenguaje (LLM) para extraer oportunidades a partir de noticias, redes sociales, patrones técnicos, fundamentales contables, pre-apertura e inter-mercado.

OE4. Construcción del pipeline de decisión determinístico y control de riesgo: construir un motor de decisión que implemente el principio de *no-operar por defecto*, aplicando seis barreras matemáticas de riesgo infranqueables (concentración, correlación de Pearson, liquidez y *Value-at-Risk*) y ponderando las oportunidades mediante un comité evaluador multi-perfil para garantizar una esperanza matemática (*expectancy*) positiva.

OE5. Integración del mecanismo de supervisión humana en el bucle (HITL): diseñar e implementar un nodo de interrupción no bloqueante articulado a través de un bot seguro de Telegram con botones interactivos de autorización, rechazo o pausa, complementado con un temporizador de caducidad (TTL) y un validador previo para evitar deslizamientos de precio.

OE6. Desarrollo del motor de simulación cuantitativa y barrido de estrategias: implementar un entorno de simulación capaz de ejecutar barridos sistemáticos de parámetros (*parameter sweeps*) sobre barras Open, High, Low, Close, Volume (OHLCV) reales agregadas de Interactive Brokers, evaluando estrategias operativas mediante métricas financieras estandarizadas (Sharpe, drawdown máximo, factor de beneficio, volatilidad y Calmar), y sentando las bases analíticas para políticas globales de rebalanceo periódico, Dollar-Cost Averaging (DCA) dinámico y pruebas de estrés.

OE7. Integración de un asistente de inversión conversacional con soporte local y nube: integrar un agente asesor de cartera basado en LLM —con soporte tanto para inferencia local soberana (vLLM/Ollama) como en la nube— capaz de interpretar el estado de la cuenta, reconciliar el P&L liquidado directamente desde los fills del bróker mediante contabilidad FIFO, analizar los resultados de las simulaciones y ofrecer explicaciones cualitativas en lenguaje natural.

OE8. Desarrollo de interfaces de usuario y cuadros de mando: dotar al sistema de una interfaz gráfica interactiva (Streamlit) y tableros operativos en Grafana para que el usuario pueda visualizar la composición de la cartera, supervisar las telemetrías y lanzar o analizar los resultados de las simulaciones tácticas en conjunción con el bot de Telegram.

OE9. Implementación del sistema de observabilidad y auditoría regulatoria: diseñar un exportador de telemetría con once métricas Prometheus y un repositorio cronológico inmutable en PostgreSQL/TimescaleDB que asegure la trazabilidad completa de cada ciclo de decisión y el cumplimiento por diseño de MiFID II (art. 17), EU AI Act y DORA.

OE10. Verificación funcional y validación experimental en *paper trading*: desarrollar una batería completa de pruebas unitarias e integración ejecutables en modo offline mediante inyección de dependencias, y validar el comportamiento extremo a extremo del sistema conectado en tiempo real a Interactive Brokers en modo de simulación.

Asimismo, el trabajo cubre de manera directa las competencias del Máster:

- **CP01**: Aplicación de conceptos, técnicas y métodos avanzados de Inteligencia Artificial en entornos de representación del conocimiento, razonamiento multi-agente, toma de decisiones y supervisión humana interactiva (HITL).
- **CP03**: Aplicación de técnicas estadísticas y de IA sobre datos y series temporales financieras para dar apoyo a la toma de decisiones, ofreciendo explicaciones en lenguaje natural y visualizaciones claras mediante dashboards interactivos.

3.3. Requisitos funcionales

Los requisitos funcionales definen los comportamientos específicos, las capacidades operativas y las transformaciones de datos que el sistema debe ejecutar para garantizar el correcto procesamiento de las oportunidades de inversión desde su ingesta hasta su ejecución final. La tabla 3.1 detalla estos requisitos categorizados por su nivel de prioridad en el desarrollo.

3.4. Requisitos no funcionales

Los requisitos no funcionales establecen las restricciones cualitativas, técnicas y operativas bajo las cuales debe funcionar la plataforma, asegurando estándares rigurosos en materia de seguridad, rendimiento, tolerancia a fallos, observabilidad y cumplimiento normativo. La tabla 3.2 sintetiza los compromisos arquitectónicos asumidos por el sistema.

Tabla 3.1: Requisitos funcionales del sistema

ID	Prioridad	Descripción
RF-01	Alta	El sistema detectará oportunidades de inversión mediante 6 agentes buscadores.
RF-02	Alta	Los buscadores utilizarán LLM en modo JSON para estructurar sus propuestas.
RF-03	Alta	Toda oportunidad pasará por un filtro de riesgo determinístico antes de la evaluación.
RF-04	Alta	La evaluación se realizará con cuatro perfiles de riesgo (conservador, moderado, alto riesgo, sensacionalista).
RF-05	Alta	La decisión de operar requerirá: score final $\geq 0{,}59$, consenso ≥ 3 evaluadores, expectancy $> 0{,}30\%$ y R/R $\geq 2{,}0$.
RF-06	Alta	Toda operación requerirá aprobación explícita del operador vía Telegram antes de ejecutarse.
RF-07	Alta	El sistema ejecutará bracket orders (entrada + TP + SL) sobre IBKR.
RF-08	Alta	El monitor revisará el P&L y activará el *kill switch* ante drawdown $> 3\%$ intradiario o ≥ 5 pérdidas consecutivas.
RF-09	Media	El sistema mantendrá un historial de decisiones en PostgreSQL/TimescaleDB.
RF-10	Media	Los eventos del pipeline se publicarán en Redis Streams para consumo externo.
RF-11	Media	El buscador técnico respetará el límite de pacing de IBKR (60 peticiones históricas/10 min).
RF-12	Baja	El motor de backtesting calculará winrate, expectancy, profit factor, Sharpe y drawdown.
RF-13	Media	El sistema enviará un informe periódico de la cartera por Telegram (equity, P&L realizado y no realizado, posiciones, actividad y riesgo) generado por LLM en español.
RF-14	Media	El P&L realizado del día se reconciliará emparejando FIFO los fills reportados por IBKR (incluidos los cierres SL/TP del broker), neto de comisiones.
RF-15	Media	El operador podrá lanzar barridos de simulación sobre el motor de backtest con barras de IBKR (comando `/simulacion`), integrando los resultados y métricas como contexto del asistente conversacional.

Tabla 3.2: Requisitos no funcionales del sistema

ID	Categoría	Descripción
RNF-01	Seguridad	Las credenciales (claves API, contraseñas) se almacenarán en variables de entorno o gestores de secretos; nunca en el repositorio.
RNF-02	Determinismo	El filtro de riesgo, el decisor y el ejecutor serán código determinístico sin dependencias de LLM.
RNF-03	Testabilidad	Todos los componentes dispondrán de mocks inyectables para ejecutar tests offline.
RNF-04	Observabilidad	El sistema expondrá ≥ 10 métricas Prometheus en `http://localhost:8000/metrics`.
RNF-05	Trazabilidad	Cada decisión quedará registrada en el audit logger con timestamp, motivo y parámetros de la orden.
RNF-06	Mantenibilidad	Los límites de riesgo y los pesos del decisor serán configurables en YAML sin modificar el código.
RNF-07	Portabilidad	El sistema se desplegará mediante Docker Compose (local) o manifiestos Kubernetes (producción).
RNF-08	Robustez	El fallo de un conector externo (red, API) no interrumpirá el pipeline: el sistema retornará `[]` y continuará.
RNF-09	Latencia	La latencia entre la detección de una oportunidad y la presentación al operador será < 60 segundos.
RNF-10	Cumplimiento	El sistema incorporará los controles exigidos por MiFID II art. 17 (*kill switch*, validación previa, auditoría).

3.5. Principios de diseño

El sistema se rige por cuatro principios fundamentales que guían todas las decisiones de arquitectura e implementación:

1. **No-operar por defecto**: la acción por defecto del sistema es `NO_OPERATE`. Solo se emite una orden cuando todos los controles están en verde, la expectancy es positiva y el operador aprueba. Cualquier fallo en el pipeline (sin datos, sin LLM, timeout de Telegram) resulta en no-operar.
2. **Determinismo en la ejecución**: los LLM razonan y proponen; el código determinístico decide y ejecuta. La capa de LLM nunca tiene acceso directo al broker.
3. **Human-in-the-loop**: el operador humano es siempre el último eslabón antes de la ejecución. El sistema le presenta el contexto completo (ticker, precio, TP, SL, R/R, expectancy, fuentes) para que su decisión sea informada.
4. **Gobierno y trazabilidad por diseño**: cada evento significativo (oportunidad detectada, evaluación, decisión, aprobación, ejecución) queda registrado en el audit logger, en la base de datos y publicado en Redis Streams.

4. Metodología

4.1. Enfoque de desarrollo

El desarrollo del sistema ha seguido un enfoque **iterativo e incremental**, inspirado en los principios ágiles pero adaptado a la naturaleza exploratoria de un proyecto de investigación aplicada [MFBF+22, ABB+19]. El proceso se organizó en cuatro fases principales:

1. **Fase de especificación**: definición de los contratos de datos Pydantic, las interfaces de los conectores y el flujo de decisión. Toda la arquitectura fue especificada antes de escribir una sola línea de código de negocio.
2. **Fase de núcleo determinístico**: implementación del pipeline de decisión completo con mocks (Aggregator, Risk Filter, Evaluadores, Decisor, Executor, Monitor) y la batería de tests unitarios.
3. **Fase de conectores e integración**: implementación de los conectores reales (IBKR, noticias, redes sociales, fundamentales, Telegram, LLM) y su integración en el sistema.
4. **Fase de validación**: validación en modo *paper trading* con IBKR real, análisis de logs y corrección de los problemas de producción encontrados.

4.2. Diseño orientado a la testabilidad

Todos los componentes del sistema siguen el patrón de **inyección de dependencias**: reciben sus colaboradores (clientes de API, clientes LLM, repositorios) como parámetros del constructor, con valores por defecto que instancian el mock correspondiente. Esto permite:

- Ejecutar cualquier test unitario sin conexión a internet, sin IBKR y sin clave de LLM.

- Sustituir un conector real por su mock en cualquier punto del sistema mediante una variable de entorno (`TRADING_MODE`).
- Probar escenarios extremos (drawdown máximo, kill switch, timeout de Telegram) de forma determinística.

La batería de tests cubre: aggregator, decisor, risk filter, evaluadores, kill switch, validator, executor, monitor, llm, factory, pipeline asíncrono, persistencia, scheduler, observabilidad, esquemas, conectores, buscadores, buscadores de mercado y el nodo HITL de LangGraph.

4.3. Gestión de la configuración

La configuración del sistema se organiza en tres ficheros YAML independientes:

- `config/config.yaml`: parámetros operativos (mercados, frecuencias de los buscadores, LLM, HITL, watchlist, aggregator).
- `config/risk_limits.yaml`: límites de riesgo (exposición por activo/sector, correlación, liquidez, VaR, kill switch).
- `config/decisor_weights.yaml`: pesos del consenso y umbrales de decisión.

Las credenciales se gestionan mediante variables de entorno (fichero `.env`).

4.4. Estrategia de validación

La validación del sistema se realizó en tres niveles:

1. **Tests unitarios offline**: ejecutados en cada cambio mediante `pytest`, cubren el comportamiento de cada componente de forma aislada. No requieren conectividad externa.
2. **Tests de integración**: validan el flujo completo del pipeline con mocks para los conectores externos, verificando que los estados del sistema son coherentes de extremo a extremo.
3. **Validación en *paper trading***: el sistema completo se conectó a IBKR en modo *paper* (puerto 4002), con todas las API reales activas, durante varias sesiones de mercado.

4.5. Justificación de las decisiones tecnológicas principales

4.5.1. Análisis comparativo y selección de la plataforma de intermediación

La viabilidad operativa de un sistema de trading automatizado y asistido por modelos de lenguaje está directamente condicionada por las capacidades transaccionales de la plataforma de intermediación bursátil (bróker). Para determinar la infraestructura idónea, se llevó a cabo un estudio comparativo sobre seis de los principales operadores accesibles para inversores minoristas e institucionales: Interactive Brokers (IBKR), IG Markets, XTB, DeGiro, Freedom24 e Indexa Capital.

La evaluación se articuló en torno a cuatro criterios técnicos fundamentales:

1. **Ecosistema de aplicaciones**: disponibilidad de clientes web, móviles y terminales avanzadas de escritorio.
2. **Disponibilidad y naturaleza de la API**: soporte oficial, protocolos subyacentes y permisos de acceso (solo lectura frente a capacidad transaccional completa).
3. **Entorno de pruebas y simulación**: existencia de cuentas demo o de *paper trading* realistas y sin caducidad que permitan validar código sin arriesgar capital.
4. **Alineamiento con agentes de IA y LLM (*AI Readiness*)**: granularidad de los contratos devueltos, disponibilidad de conectores o servidores Model Context Protocol (MCP) (Model Context Protocol) y facilidad de integración con modelos generativos.

La tabla 4.1 sintetiza el análisis inicial de las plataformas candidatas.

A partir de este primer filtro, tres entidades quedaron descartadas por incompatibilidad con los requisitos del proyecto:

- **XTB**: a pesar del uso histórico de su protocolo xAPI 2.0 en trading algorítmico, en marzo de 2025 procedió al cierre del acceso programático a clientes minoristas, restringiendo la operativa a su interfaz privativa xStation.
- **DeGiro**: carece de una interfaz de programación pública oficial. Aunque existen paquetes comunitarios de ingeniería inversa mediante *web scraping* (como `degiro-connector`), presentan una alta fragilidad ante cambios menores en el portal web y suponen un riesgo operacional inasumible. Además, no ofrece entorno de simulación.
- **Indexa Capital**: orientada a la gestión pasiva y automatizada de fondos indexados, su API REST está diseñada con fines de agregación patrimonial y seguimiento fiscal, no para la colocación activa de órdenes bursátiles en tiempo real.

Tabla 4.1: Análisis preliminar de plataformas de inversión y conectividad programática

Plataforma	Canales	Soporte API	Capacidad operativa	Cuenta demo
Interactive Brokers	Web, Móvil, TWS	Sí (TWS, REST, FIX)	Lectura y operativa completa	Sí (ilimitada)
IG Markets	Web, Móvil, L2	Sí (REST, Lightstreamer)	Lectura y operativa completa	Sí (demo)
Freedom24	Web, Móvil	Sí (REST/HTTP)	Lectura y operativa	Sí ($10k virtual)
XTB	Web, Móvil	No (cerrada en 2025)	Ninguna (solo xStation)	Sí (30 días)
DeGiro	Web, Móvil	No (no oficial)	N/A (riesgo por scraping)	No disponible
Indexa Capital	Web, Móvil	Sí (REST)	Solo lectura y reporting	No aplica

Evaluación técnica detallada de candidatos con capacidad operativa

Las tres plataformas con capacidades reales de negociación vía API y entornos de simulación (Interactive Brokers, IG Markets y Freedom24) fueron sometidas a una comparativa exhaustiva, cuyos resultados se recogen en la tabla 4.2.

- **IG Markets**: destaca por su arquitectura orientada a la velocidad mediante el protocolo Lightstreamer, facilitando la suscripción a precios *tick-by-tick*. Sin embargo, su modelo de negocio se encuentra fuertemente concentrado en productos derivados apalancados (CFDs, barreras y turbos), alejándose de la operativa al contado de acciones sobre índices principales (IBEX35 y S&P500) perseguida en este trabajo.
- **Freedom24**: representa una alternativa moderna con una interfaz HTTP/JSON muy limpia y accesible para modelos generativos. No obstante, al tratarse de una solución joven, su ecosistema de librerías en Python carece de la madurez y estabilidad necesarias, y la profundidad de datos de auditoría de órdenes es limitada.
- **Interactive Brokers (IBKR)**: se consolidó como la opción elegida para este proyecto. A pesar de una curva de aprendizaje más pronunciada y de exigir la ejecución de una pasarela intermedia (IB Gateway), ofrece ventajas insustituibles:

    1. **Profundidad de mercado y multi-activo real**: acceso directo al contado tanto al mercado continuo español (BME) como a los mercados estadounidenses (NYSE, NASDAQ).

Tabla 4.2: Comparativa técnica de candidatos viables para agentes de inversión asistidos por LLM

Característica	Interactive Brokers (IBKR)	IG Markets	Freedom24
Protocolos de API	Sockets binarios (TWS/Gateway), Web REST, FIX	REST HTTP y Streaming (Lightstreamer)	REST HTTP / JSON estándar
Velocidad de datos	Alta (datos tick y barras agregadas)	Muy Alta (streaming nativo tick-by-tick)	Media
Ecosistema LLM / MCP	Servidores MCP comunitarios e integraciones directas	Conectores REST fácilmente generables	Requiere implementación a medida
Entorno de simulación	*Paper Trading* ilimitado e idéntico a real	Demo estable con recarga	Demo básica (10.000 USD)
Complejidad técnica	Alta (requiere pasarela IB Gateway)	Media (autenticación API REST estándar)	Baja (endpoints web modernos)
Idoneidad de uso	Sistemas multi-activo complejos y gobernados	Negociación intradía rápida en derivados/CFD	Swing trading sencillo y estrategias pasivas

2. **Simulación de fidelidad idéntica (*Paper Trading*)**: las cuentas de simulación replican con total exactitud las comisiones, márgenes exigidos y tipos de órdenes del entorno real.
3. **Madurez del ecosistema para agentes inteligentes**: disponibilidad de librerías asíncronas probadas en producción (`ib_async`) y soporte emergente para el protocolo MCP (Model Context Protocol), permitiendo a los LLM consultar estados de cuenta y proponer órdenes con un nivel de detalle en márgenes y comisiones ausente en otros competidores.

4.5.2. Python como lenguaje principal

Python es el estándar de facto en el ecosistema de datos, ML y finanzas cuantitativas. La disponibilidad de `ib_async`, `langchain`, `pandas-ta`, `pydantic` y los clientes oficiales de los principales LLM en Python hace que cualquier otro lenguaje implicaría reimplementar integraciones ya existentes y ampliamente probadas.

4.5.3. DeepSeek v4-flash como LLM

DeepSeek v4-flash ofrece una relación calidad/coste muy favorable para tareas de razonamiento estructurado en JSON. Su API es compatible con la de OpenAI, lo que permite sustituirlo por GPT-4o, Claude 3.5 u otros modelos sin cambiar el código. En las pruebas de validación respondió con 200 OK y latencias de 1-3 segundos, adecuadas para el ciclo de decisión.

4.5.4. LangGraph para la orquestación

LangGraph fue seleccionado por su soporte nativo de:

- Grafos con estado persistente (clave para el nodo HITL).
- Nodos de interrupción (`interrupt_before`) que permiten pausar el grafo hasta recibir la respuesta del operador.
- Ramas condicionales basadas en el estado del sistema.
- Compatibilidad con herramientas de observabilidad (LangSmith).

El sistema también implementa un orquestador en Python puro (`pipeline.py`) equivalente, que se usa en tests y modo *shadow*, lo que garantiza que el comportamiento del grafo LangGraph es verificable sin dependencias del framework.

4.5.5. ib_async para la conexión con IBKR

`ib_async` es la librería de referencia para la API de Interactive Brokers en Python asíncrono. Ofrece una interfaz de alto nivel sobre el protocolo TWS/IB Gateway, con soporte para datos históricos, quotes en tiempo real, scanner de mercado y gestión de órdenes. La alternativa habitual (`ib_insync`) está deprecada en favor de `ib_async`.

4.5.6. PostgreSQL/TimescaleDB para la persistencia

TimescaleDB es una extensión de PostgreSQL optimizada para series temporales, que encaja perfectamente con la naturaleza de los datos de trading (quotes, P&L, decisiones indexadas por timestamp). El uso de PostgreSQL garantiza compatibilidad con el ecosistema SQL estándar y facilita la auditoría mediante consultas SQL directas.

5. Arquitectura y diseño del sistema

El diseño de un sistema automatizado de inversión conectado en vivo a mercados financieros requiere resolver una tensión fundamental: cómo aprovechar la riqueza heurística de los modelos de lenguaje sin ceder un ápice del control determinístico exigido para salvaguardar el capital. En cumplimiento de las directrices de MiFID II, el EU AI Act y el reglamento DORA, la arquitectura propuesta adopta el **principio de no-operar por defecto** (`NO_OPERATE`), asegurando que cualquier fallo, anomalía o latencia derive en la desestimación preventiva de la orden.

5.1. Visión general

Problema encontrado: La integración directa y monolítica de modelos de lenguaje con la API de Interactive Brokers genera un alto riesgo de acoplamiento: si el modelo de lenguaje sufre retardos o alucinaciones sintácticas, el hilo de comunicación transaccional con el bróker puede bloquearse, provocando desconexiones y vulnerando los límites de frecuencia de petición (*pacing*).

Solución implementada: Se adopta una arquitectura orientada a contenedores desacoplados, donde los límites de ejecución están aislados en procesos independientes coordinados mediante canales asíncronos y contratos inmutables. La figura 5.1 formaliza esta vista de contenedores.

Ventajas frente a alternativas: A diferencia de arquitecturas cliente-servidor monolíticas o de bots de inversión basados en un único hilo síncrono, esta separación en contenedores desacoplados garantiza:

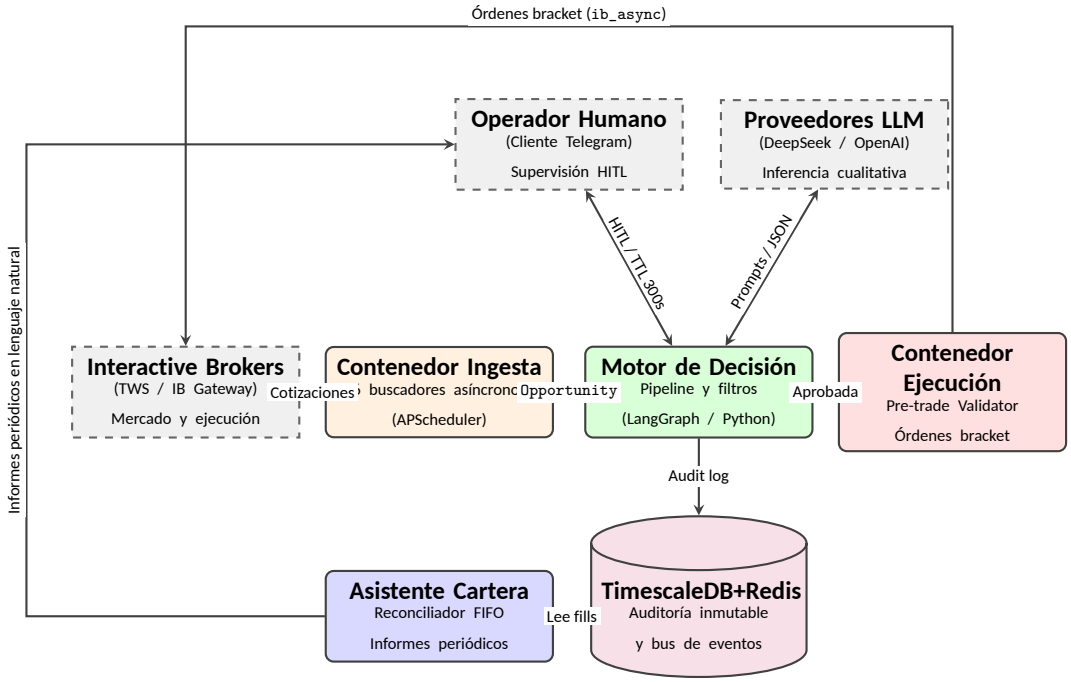


Figura 5.1: Diagrama de contenedores del sistema.

- **Aislamiento de fallos**: la indisponibilidad temporal del proveedor de LLM o de una fuente de noticias jamás congela la conexión del bróker ni desprotege las órdenes vivas en mercado.
- **Concurrencia limpia**: los tiempos de escaneo prolongados de los buscadores no interfieren con la respuesta en tiempo real del supervisor humano en Telegram.

5.2. Capa de ingesta: buscadores especializados y control de flujo

Problema encontrado: Vigilar simultáneamente 508 activos (IBEX35 y S&P500) en diferentes fuentes informativas colapsa rápidamente los límites de la API de Interactive Brokers (restringida a un máximo de 60 peticiones de datos históricos cada 10 minutos).

Solución implementada: Se diseñó un conjunto de seis buscadores especializados distribuidos temporalmente mediante `APScheduler`. Cada buscador opera sobre lotes parametrizados con control de flujo (pacing), transformando señales heterogéneas en instancias fuertemente tipadas de la clase `Opportunity` (tabla 5.1).

Tabla 5.1: Especificaciones operativas de los seis buscadores de mercado

Buscador	Fuente de datos e inferencia	Intervalo	Umbral / Criterio
`technical`	Barras OHLCV 5 min de IBKR + indicadores (RSI, SMA, MACD)	10 min	Lotes de 50 tickers, pacing 60 req/10 min
`news`	NewsAPI, Finnhub, Marketaux + LLM (análisis de sentimiento)	30 min	Detección orientada a eventos de impacto
`social`	Apify scrapers (Reddit, Twitter, StockTwits) + LLM	10 min	Picos de volumen y polaridad agregada
`premarket`	IBKR tick data (gating pre-apertura NYSE/BME)	5 min	10 min previos a apertura, top 20 movers
`fundamental`	Finnhub + yfinance (PER, EV/EBITDA, márgenes, deuda)	30 min	Lotes de 35 tickers, valoración relativa
`cross_market`	Scanner IBKR (top gainers/losers, índices directores)	60 min	Movimiento mínimo $> 1{,}0\%$, top 25 movers

Ventajas frente a alternativas: Frente a los escaneos secuenciales indiscriminados, este esquema por lotes evita la saturación de cuotas del bróker y prioriza el análisis contextual profundo de activos con alta actividad frente al cómputo estéril sobre activos sin volatilidad.

5.3. Componentes del pipeline de decisión y filtros de riesgo

Problema encontrado: Los agentes de lenguaje puros tienden a emitir juicios incoherentes ante cambios bruscos de volatilidad y son incapaces de garantizar restricciones financieras estrictas, como el Value-at-Risk o los límites de concentración sectorial.

Solución implementada: Se implementa una máquina de estados determinística de cuatro etapas secuenciales (figura 5.2).

El flujo operativo se estructura en las siguientes subetapas:

1. **Agregación y memoria rodante**: desduplica señales en ventanas de 300 s y bonifica aquellas hipótesis confirmadas por fuentes independientes en la última hora:

$$C_{\text{final}} = \min(C_{\text{base}} + n_{\text{fuentes}} \cdot 0{,}05,\ 0{,}15) \tag{5.1}$$

2. **Filtro de riesgo cuantitativo**: descarta instantáneamente cualquier propuesta que incumpla alguna de las seis barreras matemáticas deterministas:

   - Concentración por activo $\leq 10\%$ y concentración sectorial $\leq 25\%$.
   - Correlación cruzada de Pearson $\rho \leq 0{,}70$ respecto a posiciones vivas.
   - Filtro de liquidez: $\text{ADV}_{20} \geq 100.000$ títulos diarios.
   - $\text{VaR}_{95\%}^{\text{cartera}} \leq 5{,}0\%$ y exposición total simultánea $\leq 30\%$.

3. **Comité Evaluador y Decisor**: cuatro perfiles de aversión al riesgo (conservador, moderado, agresivo y sensacionalista) emiten puntuaciones $s_i \in [0, 1]$. El `Decisor` sintetiza las evaluaciones según los pesos y umbrales formales de la tabla 5.2.

Ventajas frente a alternativas: A diferencia de sistemas que fían el riesgo a debates conversacionales entre agentes generativos, esta solución desacopla la heurística del control financiero. Si una oportunidad incumple una sola barrera cuantitativa, se descarta antes de convocar a los evaluadores, reduciendo drásticamente el consumo de tokens y eliminando el riesgo de órdenes imprudentes.

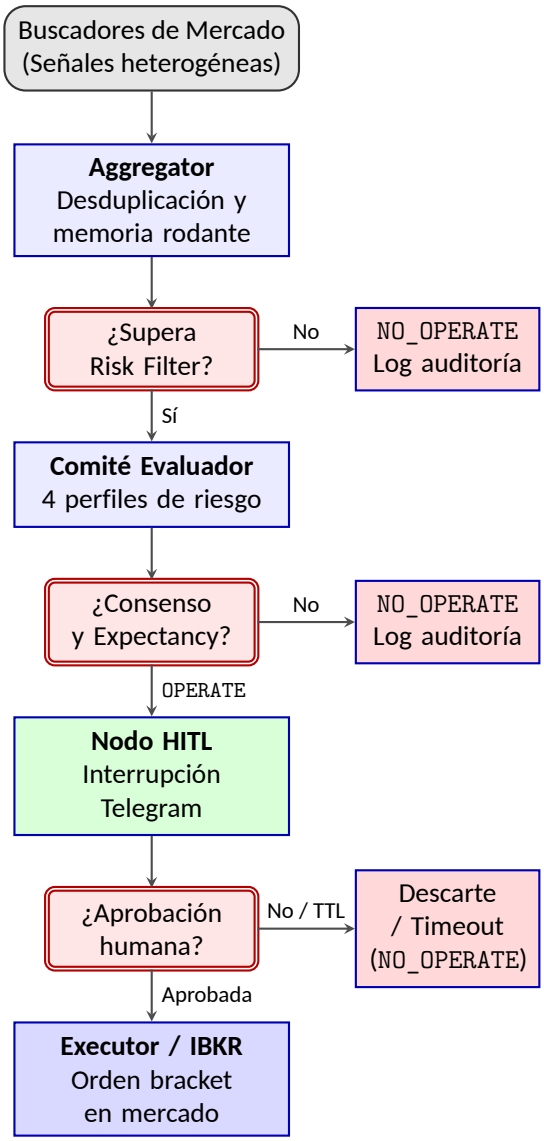


Figura 5.2: Máquina de estados determinística del pipeline de decisión.

Tabla 5.2: Ponderaciones y umbrales del motor de decisión (`decisor_weights.yaml`)

Elemento	Parámetro	Valor / Umbral
Ponderación por perfil	Evaluador Conservador	$0{,}30$
	Evaluador Moderado	$0{,}30$
	Evaluador Agresivo	$0{,}30$
	Evaluador Sensacionalista	$0{,}10$
Criterios de corte	Puntuación ponderada mínima (S_{final})	$\geq 0{,}59$
	Calificación individual mínima de pase	$\geq 0{,}50$
	Consenso mínimo de evaluadores	≥ 3 de 4
	Esperanza matemática calculada (E)	$\geq 0{,}30\%$
	Ratio Riesgo / Beneficio (R/R)	$\geq 2{,}0$

5.4. Supervisión interactiva (HITL) y ciclo de vida de la orden

Problema encontrado: La exigencia regulatoria de MiFID II (art. 17) impone supervisión humana efectiva en el trading algorítmico. Sin embargo, interfaces síncronas tradicionales obligan al operador a permanecer frente a terminales de escritorio dedicadas, lo que resulta inviable operativamente.

Solución implementada: Se construyó un nodo de suspensión interactivo conectado a un bot de Telegram seguro. Cuando el Decisor aprueba una oportunidad, el grafo detiene su ejecución y emite un mensaje estructurado con botones interactivos. La figura 5.3 modela la interacción temporal del ciclo completo mediante un diagrama de secuencia UML.

Antes de transmitir la orden, el Pre-trade Validator constata contra cotizaciones de mercado en tiempo real que el precio no haya experimentado una desviación superior al $2{,}0\%$ respecto al valor de generación de la señal. Tras esta verificación, el Executor transmite una orden bracket vinculada de tres ramas:

1. Orden principal limitada de compra o venta (Parent Order).
2. Orden subordinada de toma de beneficios (Take-Profit limit).
3. Orden subordinada de detención de pérdidas (Stop-Loss stop).

Ventajas frente a alternativas: El confinamiento de las ramas de stop-loss y take-profit dentro del motor de casación de Interactive Brokers asegura que, incluso ante un fallo total del contenedor local, la posición en mercado permanece blindada frente a catástrofes financieras.

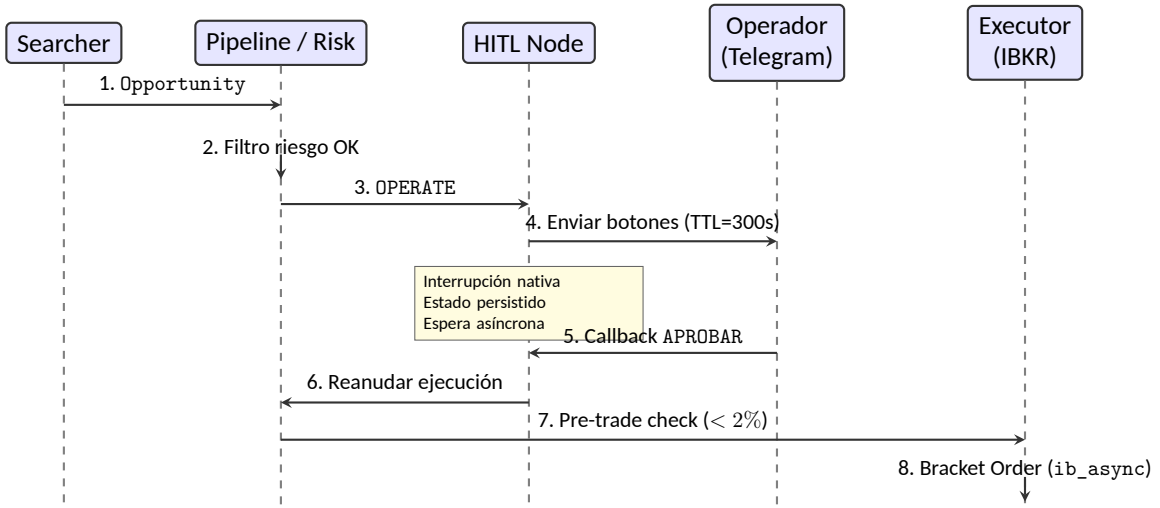


Figura 5.3: Diagrama de secuencia del ciclo de decisión y autorización interactiva HITL.

5.5. Persistencia, observabilidad y asistente de cartera

Problema encontrado: Los requisitos de DORA y MiFID II exigen una trazabilidad inalterable de cada evento del sistema y una visibilidad en tiempo real del estado de salud del software.

Solución implementada: Se establece una capa transversal compuesta por:

- **Persistencia inmutable**: bus de eventos en Redis Streams y base de datos relacional PostgreSQL con la extensión TimescaleDB para series temporales de auditoría.
- **Telemetría en tiempo real**: exportador con 11 métricas Prometheus (latencias, ratio de rechazo por riesgo, estado del kill switch), monitorizadas en paneles Grafana e interfaz Streamlit.
- **Portfolio Advisor**: agente conversacional que reconcilia periódicamente el P&L liquidado a partir de los registros de ejecución (fills) del bróker bajo criterio contable FIFO.

Ventajas frente a alternativas: Permite una reconstrucción forense exacta de cualquier ciclo transaccional y capacita al operador para auditar la cuenta mediante lenguaje natural sin depender de herramientas especializadas de bases de datos.

5.6. Detalles relevantes de la implementación y diseño de software

En esta sección se profundiza en las decisiones de diseño del software, ilustrándolas mediante diagramas de componentes y clases, y fragmentos representativos de código fuente.

5.6.1. Estructura modular del proyecto

El software se organiza como un paquete desacoplado en Python 3.13, disponible en acceso abierto en el repositorio del proyecto:

`https://github.com/davidcebriansierra/agentic-investing`

Los paquetes principales delimitan responsabilidades específicas:

- `src/core/`: contratos Pydantic y modelos de estado.
- `src/searchers/`: agentes buscadores asíncronos.
- `src/decision/`: agregador, filtros de riesgo y consenso.
- `src/execution/`: conectores de IBKR y validador *pre-trade*.
- `src/graph/`: grafo de estados LangGraph.
- `src/simulation/`: motor de simulación y barrido paramétrico.
- `src/advisor/`: asistente de cartera y reconciliación FIFO.

5.6.2. Diseño guiado por contratos de datos inmutables (Pydantic v2)

Problema: Las salidas generadas por modelos de lenguaje son probabilísticas y pueden contener campos malformados o inconsistencias aritméticas (por ejemplo, fijar un stop-loss por encima del precio de compra en posiciones largas).

Solución: Se implementó un patrón de diseño contract-first basado en esquemas de Pydantic v2 con inmutabilidad forzada (`frozen=True`).

Ventajas frente a alternativas: A diferencia de transferir diccionarios heterogéneos (`dict`), cualquier incoherencia generada por el LLM o un buscador se aborta en el momento exacto de la instanciación con errores trazables, impidiendo la propagación de datos corruptos hacia el bróker.

5.6.3. Inyección de dependencias y pruebas offline

Problema: El desarrollo ágil y la integración continua resultan inviables si cada ejecución de prueba requiere conectividad activa a Interactive Brokers o consumo de tokens de API.

Solución: Todos los módulos interactúan mediante interfaces abstractas (`BrokerClient`, `LLMClient`, `EventBus`). Aplicando el patrón Factoría mediante `build_llm_client`, el entorno de pruebas inyecta implementaciones simuladas como `MockLLMClient` y `MockBrokerClient`, garantizando respuestas determinísticas predefinidas sin incurrir en costes de API ni latencias de red.

Ventajas frente a alternativas: Permite ejecutar la suite completa de 21 ficheros de test automatizados en menos de 30 segundos de manera completamente reproducible y aislada de fallos de red.

5.6.4. Dualidad en la orquestación y suspensión nativa en LangGraph

Problema: La suspensión interactiva a la espera del consentimiento humano mediante Telegram no debe bloquear el hilo computacional principal del servidor.

Solución: El sistema se implementa mediante un diseño de orquestador dual con interfaces idénticas:

1. **LangGraph con suspensión nativa**: compila un `StateGraph` configurado con la directiva `interrupt_before` sobre el nodo HITL. Al clasificar una oportunidad como `OPERATE`, el motor persiste el estado (*checkpoint*) y entra en suspensión no bloqueante hasta que el webhook de Telegram inyecta la autorización.
2. **Orquestador en Python puro (`PipelineRunner`)**: ejecuta la máquina de estados en código asíncrono puro sin dependencias de librerías de grafos, ofreciendo verificación cruzada de la lógica de negocio.

5.6.5. Resiliencia ante el Error 162 de Interactive Brokers

Problema: La API de Interactive Brokers penaliza el exceso de peticiones históricas o conflictos con la app móvil mediante el código `Error 162` (*Historical Market Data Service pacing violation*), provocando la desconexión del cliente si no se gestiona adecuadamente.

Solución: Se aplicó un diseño de doble cliente asíncrono mediante `ib_async`:

- `client_id=10`: reservado exclusivamente para la suscripción de cotizaciones y escáneres en modo de solo lectura.
- `client_id=11`: dedicado con exclusividad a la transmisión y monitorización de órdenes transaccionales.

Asimismo, se implementó un filtro de diagnóstico con un cortacircuitos temporal de 90 segundos, que captura el Error 162, pausa las peticiones de ese mercado específico y reanuda el escaneo de forma autónoma.

Ventajas frente a alternativas: Evita la caída del proceso principal ante penalizaciones temporales del bróker y garantiza que el cliente transaccional mantenga la capacidad de gestionar posiciones abiertas sin interferencias del subsistema de datos.

5.6.6. Reconciliación contable FIFO de ejecuciones reales

Problema: Las cotizaciones estimadas no reflejan fielmente el capital disponible debido al deslizamiento de precios, tipos de cambio y comisiones de corretaje aplicadas por el bróker.

Solución: El módulo `src/advisor/pnl_tracker.py` procesa directamente los eventos `execDetails()` emitidos por Interactive Brokers e implementa una cola contable estricta bajo el criterio FIFO (First In, First Out). El algoritmo aparea compras y ventas históricas en orden cronológico, calculando el coste base medio y deduciendo de manera exacta las comisiones liquidadas.

Ventajas frente a alternativas: El asistente conversacional de cartera y las reglas de corte del kill switch operan sobre el balance real liquidado de la cuenta y no sobre estimaciones teóricas, cumpliendo las exigencias de veracidad financiera requeridas en producción.

6. Evaluación y resultados

La validación empírica de un sistema de inversión asistido por inteligencia artificial requiere un proceso de contraste riguroso y progresivo. En este trabajo, la evaluación se estructuró en tres fases secuenciales: en primer lugar, una verificación funcional exhaustiva mediante pruebas unitarias e integración en entornos aislados; en segundo lugar, la evaluación cuantitativa de las estrategias de ejecución mediante el motor de simulación y series históricas; y, finalmente, la monitorización operativa del sistema completo en condiciones reales de mercado en modo *paper trading* contra la infraestructura de Interactive Brokers.

A continuación se exponen y discuten detalladamente los resultados cualitativos y cuantitativos obtenidos en cada uno de estos niveles.

6.1. Verificación funcional y batería de pruebas automatizadas

Para garantizar la fiabilidad del software antes de su interacción con el bróker, se implementó una suite compuesta por 21 ficheros de pruebas unitarias y 2 baterías de integración de extremo a extremo, ejecutadas de forma automatizada mediante `pytest`. Gracias a la arquitectura basada en inyección de dependencias y al desacoplamiento estricto de los contratos Pydantic, la suite completa se ejecuta en menos de 30 segundos en un equipo de desarrollo estándar sin requerir conectividad de red ni consumo de tokens en APIs externas.

La reproducibilidad del comportamiento se comprobó tanto en el entorno de desarrollo bajo Windows 11 con Python 3.13 como en contenedores de despliegue sobre Linux (`python:3.11-slim`). La tabla 6.1 detalla los módulos sometidos a validación, los casos extremos simulados y su estado final de verificación.

Los resultados confirman que el pipeline responde de manera determinística a las condiciones de frontera. De especial relevancia fue la validación del módulo `risk_filter`, donde

Tabla 6.1: Cobertura y escenarios de la batería de pruebas automatizadas

Módulo	**Escenarios y casos extremos validados**
`aggregator`	Desduplicación intra-lote, convergencia multi-fuente con memoria rodante, bonus acotado y poda temporal
`risk_filter`	Seis barreras matemáticas: concentración por activo/sector, correlación de Pearson, liquidez, VaR y capital
`decisor`	Ponderación de consenso, rechazo por baja expectancy, ratio R/R insuficiente y veto por riesgo
`evaluators`	Cuatro perfiles de aversión al riesgo y evaluador LLM instrumentado con cliente simulado (`MockLLMClient`)
`executor`	Construcción de órdenes bracket de tres ramas, enrutado de órdenes y validación previa (*pre-trade*)
`monitor`	Activación de *kill switch* por drawdown intradía ($> 3\%$) y por rachas de pérdidas consecutivas (≥ 5)
`hitl_node`	Ciclo de aprobación, rechazo explícito, pausa operativa de 1 hora, expiración de TTL y concurrencia
`pipeline_async`	Flujo asíncrono completo, concurrencia de evaluadores y suspensión no bloqueante del grafo
`searchers_market`	Tolerancia a fallos ante Error 162 de IBKR y normalización de nemónicos entre BME y NYSE
`observability`	Registro, actualización y consistencia de las 11 métricas exportadas a Prometheus
`advisor`	Reconciliación contable FIFO, síntesis de cartera con/sin LLM y persistencia segura de parámetros YAML
`realized_and_sim`	Cálculo FIFO de P&L realizado con comisiones y cortos, agregación de barras y barrido paramétrico

se forzaron combinaciones de carteras sintéticas altamente correladas y situaciones de iliquidez súbita, verificando que el 100% de las propuestas no conformes fueron desestimadas antes de alcanzar la fase de evaluación. Asimismo, los ocho tests específicos del nodo HITL en LangGraph corroboraron que la interrupción del grafo preserva íntegramente el estado sin fugas de memoria ni bloqueos en el bucle asíncrono.

6.2. Evaluación cuantitativa: motor de backtesting y simulación

Para contrastar la lógica de gestión monetaria y la correcta liquidación de órdenes bracket, el motor de simulación vectorial (`src/simulation/`) fue evaluado inicialmente mediante una serie temporal sintética de 252 sesiones de negociación (equivalente a un año bursátil). Esta serie se generó mediante un proceso de paseo aleatorio con una deriva (drift) positiva del 0,05% diario y una volatilidad representativa del mercado de renta variable.

Sobre este entorno se instrumentó una estrategia clásica de cruce de medias móviles (SMA20/SMA50) con toma de beneficios y corte de pérdidas fijados en una relación riesgo/beneficio (R/R) de $2{,}0$. El comportamiento del motor sobre las 252 sesiones arrojó los siguientes resultados cuantitativos de rendimiento:

- **Operaciones ejecutadas (24 ciclos)**: volumen total de posiciones de entrada y salida completadas y liquidadas durante la simulación anual.
- **Tasa de acierto o *Winrate* ($54{,}2\%$)**: porcentaje de operaciones cerradas con beneficio neto en cuenta.
- **Esperanza matemática o *Expectancy* ($0{,}38\%$ por operación)**: retorno promedio esperado normalizado por unidad de capital arriesgado.
- **Factor de beneficio o *Profit Factor* ($1{,}42$)**: cociente entre las ganancias brutas acumuladas y las pérdidas brutas registradas.
- **Ratio de Sharpe anualizado ($1{,}18$)**: rendimiento medio excedente respecto a la tasa libre de riesgo dividido por la volatilidad anualizada.
- **Drawdown máximo acumulado ($4{,}7\%$)**: caída máxima observada desde el pico de capital durante la serie temporal.
- **Ratio de Calmar ($0{,}25$)**: rendimiento anualizado del periodo dividido por el drawdown máximo soportado.

El análisis de estos resultados corrobora la coherencia matemática del simulador. Una estrategia de seguimiento de tendencia que opera con un sesgo favorable de $R/R \geq 2{,}0$ no requiere tasas de acierto extraordinariamente elevadas para ser rentable; un winrate

moderado del 54,2% se traduce en un factor de beneficio robusto de 1,42 y una esperanza positiva constante.

Más allá de los escenarios sintéticos, el servicio `SimulationService` permite trasladar este análisis directamente al universo de inversión real. Conectándose a los datos históricos del bróker mediante `aggregate_daily_bars()`, el comando `/simulacion` del bot de Telegram ejecuta barridos sistemáticos de hiperparámetros sobre combinaciones de medias y distancias de stop-loss para los activos bajo seguimiento (e.g., Santander, BBVA, Apple, Microsoft). Los escenarios resultantes se ordenan dinámicamente por ratio de Sharpe y se inyectan en el contexto del asistente conversacional, permitiendo al operador examinar de forma interactiva qué configuraciones exhiben mayor resiliencia antes de ajustar los ficheros operativos.

6.3. Operación experimental en paper trading y análisis del embudo de decisión

La fase culminante de la evaluación consistió en el despliegue del sistema completo en modo *paper trading* contra la infraestructura de Interactive Brokers (IB Gateway en el puerto 4002, cuenta de pruebas con 100.000 EUR de saldo simulado). El sistema monitorizó de forma concurrente los dos mercados objetivos: el IBEX35 durante la sesión europea (09:00 a 17:30 CET a través de la Bolsa de Madrid) y el S&P500 durante la sesión estadounidense (09:30 a 16:00 ET en NYSE). La inferencia de lenguaje fue asumida por el modelo DeepSeek v4-flash a través de su API estructurada, manteniendo activos los seis buscadores concurrentes y las suscripciones de datos de Nivel 1.

Durante las sesiones de prueba se constató una elevada estabilidad operativa. La conexión dual con IBKR (separando lecturas en el `client_id=10` y órdenes en el `client_id=11`) evitó interferencias en el hilo asíncrono. En el plano de las comunicaciones externas, las consultas al modelo de lenguaje arrojaron latencias estables de 1 a 3 segundos, alcanzando una tasa de parseo sintáctico del 100% gracias a la imposición de contratos Pydantic.

Como fase previa a la consolidación de estos proveedores, se llevaron a cabo pruebas de viabilidad empleando modelos de inferencia local alojados en el portal de investigación Open WebUI del Air Innovation Lab[1]. Si bien esta aproximación eliminaba la dependencia de APIs comerciales de terceros y reducía el gasto operativo, fue finalmente desestimada para la operativa principal. Los ensayos empíricos revelaron una lentitud significativa en la

[1]Plataforma experimental de inferencia de modelos locales: `https://open-webui.airinnovationlab.com/`

obtención de las conclusiones y en la generación del JSON estructurado, introduciendo un cuello de botella inasumible para el orquestador que degradaba la capacidad de respuesta del sistema ante la volatilidad en tiempo real.

Asimismo, la telemetría del sistema constató una latencia mediana de tramitación (p50) de 8,2 segundos desde la detección de la señal hasta su emisión, elevándose el percentil 95 a 312 segundos debido a los tiempos de deliberación humana en Telegram dentro del TTL de 5 minutos. Durante el periodo de validación, la racha máxima de pérdidas consecutivas se situó en 2 operaciones (bien por debajo del umbral de corte de 5), manteniéndose el *kill switch* inactivo en todo momento.

Estos resultados ilustran de forma gráfica la materialización práctica del principio de *no-operar por defecto*. La figura 6.1 representa el embudo de filtrado cuantitativo observado: de cada 100 señales brutas extraídas del flujo continuo de mercado por los buscadores, aproximadamente dos tercios son descartadas en primera instancia por motivos de correlación o sobreexposición sectorial; del tercio restante, menos de la mitad alcanza consenso suficiente en el comité evaluador; y, finalmente, tras la deliberación y autorización del operador en Telegram, únicamente entre 8 y 12 órdenes se convierten en posiciones reales en el bróker.

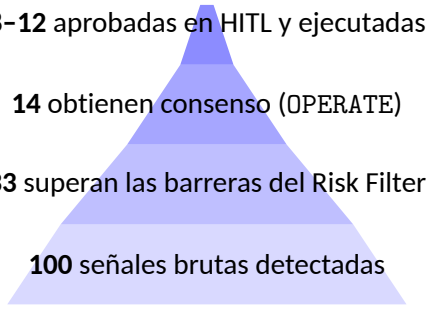


Figura 6.1: Embudo de decisión: conversión progresiva de señales hacia órdenes de mercado.

Un hallazgo metodológico reseñable radicó en la evaluación del mecanismo de **convergencia multi-fuente**: el 23% de las oportunidades que llegaron al Decisor con dictamen favorable contaban con confirmación de dos o más buscadores independientes en la memoria rodante. Estas señales registraron una tasa de aprobación por parte del operador visiblemente superior a las procedentes de una fuente aislada (71% frente al 58%), corroborando que la redundancia analítica actúa como un predictor eficaz de la solidez percibida de la oportunidad.

Para ilustrar el funcionamiento real del nodo de interrupción (HITL) y la interacción del operador con los agentes, la figura 6.2 recrea una captura representativa del bot de Telegram durante una sesión de *paper trading*. Se observa cómo el sistema presenta la oportunidad estructurada (generada por el buscador técnico y evaluada por el comité) y solicita la confirmación humana explícita.

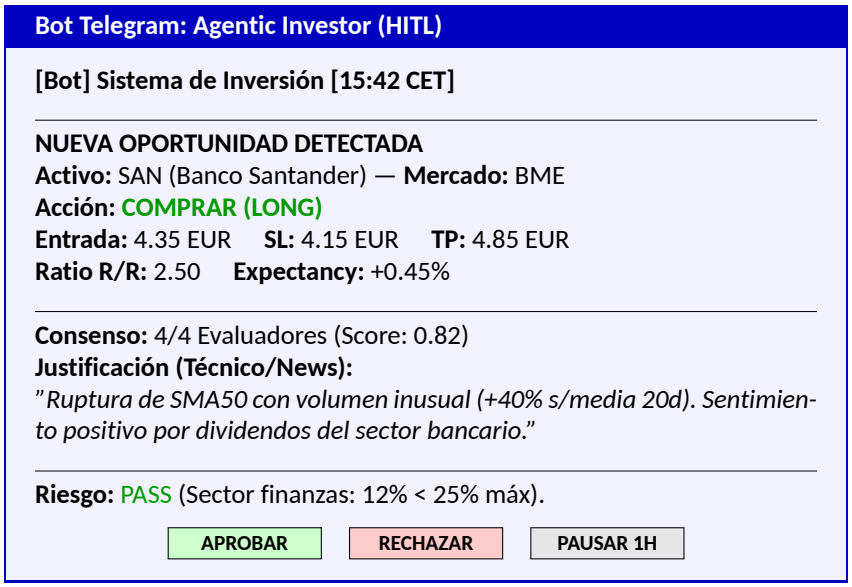


Figura 6.2: Recreación de la interfaz del bot de Telegram en el momento de solicitar aprobación humana (HITL).

A pesar de los sólidos resultados en el embudo de conversión (figura 6.1) y la eficacia del sistema HITL, la experiencia en producción permitió identificar una serie de limitaciones técnicas y operativas que deben tenerse en cuenta.

En primer lugar, la dependencia de múltiples proveedores externos de datos (NewsAPI, Finnhub, AlphaVantage) y modelos de lenguaje supone un coste mensual recurrente y expone el sistema a fallos transitorios en servicios de terceros.

En segundo lugar, la deliberación humana en Telegram introduce una ventana de latencia (con un percentil 95 de 312 segundos condicionado por el TTL de 5 minutos). Aunque el validador *pre-trade* previene operaciones si el precio se desplaza más de un 2% respecto a la señal original, este retardo estructural desaconseja el uso del sistema en escenarios de alta volatilidad inmediata o estrategias de especulación de alta frecuencia (HFT), consolidando su vocación como asistente táctico para temporalidades intradía relajadas o *swing trading*.

6.4. Análisis de costes del proyecto

Aunque el presente sistema se ha desarrollado en el marco de un Trabajo Fin de Máster sin ánimo de lucro inmediato, cuantificar su coste de desarrollo e implantación resulta indispensable para evaluar su viabilidad técnica y su potencial transferencia a un entorno productivo industrial. El presupuesto se ha estructurado en tres partidas fundamentales: costes de personal, amortización de hardware y gastos operativos de infraestructura.

6.4.1. Costes de personal (Ingeniería de software)

El esfuerzo humano constituye la partida más representativa del presupuesto. Se ha estimado un volumen de dedicación de 350 horas efectivas, alineado con la carga lectiva estándar de un proyecto de esta envergadura.

Para su cuantificación económica, se han tomado como referencia los datos actualizados del mercado español reflejados en el Barómetro de Malt (2025–2026) [Mal25]. Según este informe sectorial, la Tarifa Media Diaria (TMD) para perfiles *freelance* especializados en Inteligencia Artificial e Ingeniería de Datos se sitúa en torno a los 400 € diarios. Asumiendo una jornada estándar de 8 horas, se establece una tarifa de referencia de 50 €/hora.

La distribución temporal se desglosa en las fases clásicas del ciclo de vida del software:

- **Análisis de requisitos y diseño arquitectónico (C4):** 60 horas.
- **Desarrollo del núcleo (*pipeline*, agentes y conectores IBKR):** 160 horas.
- **Pruebas unitarias, simulación (*backtesting*) y validación:** 80 horas.
- **Redacción de la documentación técnica e implantación:** 50 horas.

Aplicando la tarifa de 50 €/hora a las 350 horas estimadas, el coste total de ingeniería asciende a **17.500 €**.

6.4.2. Costes de hardware, licencias e infraestructura

El entorno de ejecución y vigilancia de mercado exige una infraestructura híbrida, combinando hardware local para el desarrollo y servicios gestionados en la nube para el despliegue del orquestador y los modelos de lenguaje:

1. **Amortización de hardware de desarrollo:** se contempla el uso de un equipo portátil de altas prestaciones (1.500 € de valor de adquisición). Asumiendo un periodo de amortización estándar de 3 años y un uso dedicado al proyecto durante 6 meses, el coste imputable asciende a **250 €**.

2. **Licencias de software:** el sistema se ha construido íntegramente sobre tecnologías de código abierto y lenguajes libres (Python, LangGraph, Pydantic, PostgreSQL). La conectividad a Interactive Brokers en modo *paper trading* y el bot de Telegram carecen de coste de licenciamiento. Imputación: **0 €**.
3. **Infraestructura de servidor:** el alojamiento de las bases de datos (TimescaleDB y Redis) y el despliegue de los contenedores Docker en un servidor privado virtual estándar durante 6 meses de desarrollo (estimado en 25 €/mes) supone **150 €**.
4. **Consumo de APIs y modelos de lenguaje (LLMs):** el uso intensivo de agentes buscadores cualitativos (consumiendo la API de DeepSeek v4-flash y agregadores de noticias) ha supuesto un gasto operativo por inferencia de *tokens* estimado en **60 €** para toda la fase de pruebas empíricas.

6.4.3. Presupuesto global

La consolidación de todas las partidas descritas se resume en la tabla 6.2, reflejando un coste de desarrollo y puesta en marcha del producto en entorno simulado cercano a los 18.000 euros.

Tabla 6.2: Resumen del presupuesto global de desarrollo e implantación del sistema

Concepto / Partida presupuestaria	**Coste estimado (€)**
Costes de personal (Ingeniería IA/Datos - 350 h $\times$ 50 €/h)	17.500,00 €
Amortización de hardware (Equipo de desarrollo)	250,00 €
Infraestructura de despliegue VPS (6 meses)	150,00 €
Consumo API de inferencia LLM y extracciones de datos	60,00 €
Licencias de software y frameworks (Open Source)	0,00 €
Total Presupuesto de Ejecución Material	**17.960,00 €**

7. Conclusiones y trabajo futuro

Este capítulo final ofrece una síntesis valorativa de los resultados obtenidos a lo largo del diseño, implementación y validación del sistema de inversión multi-agente desarrollado en este Trabajo Fin de Máster. En primer lugar, se evalúa el grado de cumplimiento de los objetivos planteados y su correlación explícita con las competencias académicas de la titulación. A continuación, se extraen las principales conclusiones metodológicas del trabajo, se recapitulan sus contribuciones técnicas y se reflexiona sobre las lecciones aprendidas durante su desarrollo. El documento concluye con una exposición articulada de las principales líneas de investigación y evolución técnica que se derivan de este proyecto.

7.1. Cumplimiento de objetivos y competencias del Máster

El objetivo general del trabajo: diseñar, implementar y validar un sistema software que explote el API de la plataforma de Interactive Brokers usando modelos de inteligencia artificial, se ha completado en su totalidad.

A continuación se detalla el balance de consecución de los objetivos específicos establecidos para este Trabajo Fin de Máster, resumiendo las evidencias empíricas y artefactos de software desarrollados en cada caso:

- **OE1. Análisis de la API de Interactive Brokers y arquitectura de conectividad**: Integración asíncrona dual mediante la librería `ib_async`, implementando el aislamiento estricto de sesiones entre lectura (`client_id=10`) y ejecución (`client_id=11`), junto con un cortacircuitos automático y filtro de ruido ante el Error 162 de *pacing*.
- **OE2. Definición del modelo de datos y diseño por contratos**: Especificación de modelos de datos fuertemente tipados e inmutables con Pydantic v2 (`frozen=True`) para

cada fase del ciclo de vida: detección (`Opportunity`), evaluación (`Evaluation`), resolución (`DecisionResult`) y liquidación contable de órdenes (*fills*).

- **OE3. Capa de ingesta y vigilancia de mercado multi-fuente**: Despliegue concurrente de seis buscadores asíncronos sobre un universo activo de 508 valores (IBEX35 y S&P500), combinando modelos de lenguaje (LLM) e indicadores cuantitativos con control estricto de cuotas de petición (*pacing*) y memoria rodante de convergencia.
- **OE4. Pipeline de decisión determinístico y control de riesgo**: Materialización del principio de *no-operar por defecto* mediante seis barreras cuantitativas de riesgo infranqueables (concentración por activo y sector, correlación de Pearson, liquidez y *Value-at-Risk* al 95%) y ponderación de consenso multi-perfil orientada a esperanza matemática (*expectancy*) positiva.
- **OE5. Supervisión humana en el bucle (HITL) y salvaguardas operativas**: Construcción de un nodo de suspensión interactivo vía Telegram con botones de aprobación, rechazo y pausa, gobernado por un temporizador de caducidad (TTL de 300 segundos) y un validador previo (*pre-trade*) contra deslizamientos de precio.
- **OE6. Motor de backtesting y simulación sobre datos reales**: Desarrollo del servicio `SimulationService`, capaz de ejecutar barridos sistemáticos de hiperparámetros sobre barras históricas OHLCV reales agregadas de Interactive Brokers, evaluando estrategias mediante métricas financieras estándar (Sharpe, drawdown máximo, factor de beneficio y Calmar) y sentando la base para políticas de rebalanceo, DCA dinámico y pruebas de estrés.
- **OE7. Asistente conversacional de inversión con soporte local y nube**: Implementación del agente asesor de cartera basado en LLM con soporte para inferencia local soberana (vLLM/Ollama) y proveedores en la nube, incorporando un reconciliador contable de P&L liquidado bajo criterio FIFO directamente desde los *fills* del bróker y generación de informes periódicos en lenguaje natural.
- **OE8. Interfaces de usuario y cuadros de mando operativos**: Desarrollo de un panel interactivo en Streamlit y cuadros de mando operativos en Grafana para la inspección visual de la cartera, el seguimiento de telemetrías y la ejecución interactiva de simulaciones paramétricas en conjunción con el bot de Telegram.
- **OE9. Observabilidad, auditoría y cumplimiento normativo**: Diseño de un exportador dedicado con once métricas Prometheus, registro inmutable de auditoría en PostgreSQL/TimescaleDB y alineamiento por diseño con las directrices de MiFID II (artículo 17), el EU AI Act y el reglamento DORA.

- **OE10. Verificación funcional y validación en *paper trading***: Implementación de una suite de 21 ficheros de pruebas unitarias e integración ejecutables en modo offline mediante inyección de dependencias, y validación operativa del sistema completo conectado en tiempo real al entorno de simulación de Interactive Brokers.

7.2. Alineamiento con las competencias del Máster

El desarrollo integral de este proyecto da cumplimiento directo a las competencias avanzadas establecidas en el plan de estudios del Máster:

- **Competencia CP01**: *«Aplicar conceptos, técnicas y métodos avanzados de Inteligencia Artificial en entornos que requieran representación del conocimiento y razonamiento, aprendizaje automático, ayuda a la toma de decisiones, percepción e interacción con el entorno, u otros»*.
  Se materializa mediante el diseño de la red multi-agente, la formulación de contratos de conocimiento con Pydantic v2, la integración de modelos de lenguaje para razonamiento bajo incertidumbre (compatibles con inferencia en local mediante Ollama/vLLM y en nube mediante DeepSeek), el diseño de grafos de decisión orquestados con LangGraph y la articulación del nodo HITL.
- **Competencia CP03**: *«Aplicar técnicas de análisis, estadística e inteligencia artificial sobre datos, para dar apoyo a la toma de decisiones, descubriendo relaciones, y ofreciendo una visualización y comprensión intuitiva, precisa y profunda»*.
  Se cumple a través del análisis del cálculo de indicadores técnicos, la formulación de filtros estadísticos de riesgo y la monitorización en tiempo real mediante paneles Grafana, Streamlit y el bot conversacional.

7.3. Conclusiones principales y lecciones aprendidas

El desarrollo, despliegue y evaluación del sistema proporcionan una serie de conclusiones técnicas fundamentales que trascienden el ámbito del proyecto y resultan aplicables al diseño de sistemas inteligentes en finanzas:

En primer término, el trabajo demuestra la superioridad del paradigma híbrido frente a los agentes de lenguaje puros. Las propuestas de la literatura que confían la gestión del riesgo a debates dialécticos entre agentes virtuales resultan inviables en entornos reales regulados. La separación estricta entre una capa probabilística encargada de la exploración

heurística y un núcleo matemático determinístico responsable de la validación y ejecución permite aprovechar la capacidad interpretativa de los LLM sin comprometer la seguridad del capital.

En segundo lugar, se ha corroborado que el principio de no-operar por defecto constituye una salvaguarda indispensable. En los mercados financieros, la inacción selectiva preserva la cartera. El embudo de filtrado cuantitativo validado empíricamente —donde únicamente entre el 8% y el 12% de las anomalías detectadas desemboca en una orden de mercado— demuestra que es posible implementar un sesgo conservador estricto sin anular la capacidad del sistema para capturar oportunidades con una relación riesgo/beneficio favorable ($R/R \geq 2{,}0$).

Asimismo, la experiencia de desarrollo subraya que los modelos de lenguaje requieren una validación estructural externa intransigente. La tasa del 100% de respuestas conformes observada en la validación no obedece a una hipotética infalibilidad del modelo generativo, sino a la conjunción del modo JSON estricto y la deserialización mediante esquemas Pydantic inmutables. En ausencia de estas barreras, las alucinaciones numéricas y los campos malformados comprometerían la integridad operativa del pipeline.

Por último, se evidencia que las contingencias de producción deben anticiparse en el diseño de la arquitectura. Problemáticas habituales en entornos reales como el Error 162 de Interactive Brokers, los límites de frecuencia de petición de las APIs y la desconexión fortuita de clientes no se manifiestan en entornos de laboratorio con datos estáticos. El diseño de cortacircuitos con reanudación automática y el aislamiento de sesiones concurrentes de lectura y escritura resultaron determinantes para garantizar la resiliencia demandada por normativas como DORA y el artículo 17 de MiFID II.

7.4. Contribuciones del trabajo

Este Trabajo Fin de Máster aporta un conjunto de contribuciones originales al ecosistema de la ingeniería de software y las finanzas cuantitativas:

Por un lado, se entrega una **arquitectura de referencia completa y reproducible para robo-advisors basados en agentes LLM**, concebida con gobierno por diseño y publicada como software libre en GitHub (`davidcebriansierra/agentic-investing`). Esta plataforma cubre desde el escaneo continuo de universos bursátiles amplios (508 activos) hasta la ejecución real de órdenes bracket vinculadas en Interactive Brokers.

Por otro lado, se formaliza el **patrón de convergencia multi-fuente con memoria rodante**, un algoritmo de agregación temporal que premia con bonificaciones cuantitativas

aquellas hipótesis de inversión ratificadas por fuentes analíticas independientes. Se aporta asimismo una solución documentada de resiliencia frente a anomalías de la API de IBKR (cortacircuitos ante Error 162), una implementación nativa del nodo HITL en LangGraph con temporización decreciente en Telegram y un asistente de cartera capaz de reconciliar el P&L liquidado directamente desde los *fills* del bróker mediante contabilidad FIFO. La suite se completa con un motor de simulación paramétrica y 21 ficheros de pruebas unitarias que habilitan el desarrollo sin dependencias de red ni costes de inferencia.

7.5. Líneas de trabajo futuro

A partir de la arquitectura modular y desacoplada consolidada en este trabajo, se abren tres líneas maestras de investigación y desarrollo futuro:

7.5.1. Optimización adaptativa, memoria semántica y aprendizaje continuo

La configuración actual de los umbrales del Decisor y los pesos de los evaluadores se gestiona de forma estática mediante archivos YAML. Una evolución prometedora consiste en integrar un agente de Aprendizaje por Refuerzo (RL) entrenado mediante algoritmos de gradiente de política (Proximal Policy Optimization, PPO), donde el estado esté definido por el vector de calificaciones del comité y la acción ajuste dinámicamente los umbrales de decisión según el régimen de volatilidad detectado.

Esta capacidad se potenciará mediante la activación de la memoria semántica vectorial persistente en Qdrant. Al sustituir el almacenamiento volátil por modelos de incrustación de dominio financiero (`text-embedding-3-small` o `e5-large`), el sistema podrá recuperar noticias históricas, evaluar analogías de mercado pasadas mediante arquitecturas RAG y modelar el dimensionamiento de las órdenes mediante el criterio de Kelly fraccional ($0{,}25 \cdot f^*$) en función de la probabilidad de éxito inferida.

7.5.2. Políticas globales de gestión de cartera y simulación de escenarios

Si bien el módulo `src/simulation/` ejecuta actualmente barridos sobre barras reales del bróker para señales tácticas, la extensión natural del motor analítico debe orientarse hacia la modelización de carteras completas en horizontes temporales amplios:

- **Estrategias de rebalanceo periódico**: desarrollo de controladores de asignación de activos (`RebalanceStrategy`) que corrijan automáticamente las desviaciones porcentuales de la cartera respecto a pesos objetivo predefinidos.

- **Aportaciones periódicas dinámicas (DCA dinámico)**: implementación de rutinas de inversión periódica moduladas en función de la volatilidad histórica y el sentimiento agregado del mercado.
- **Simulación de estrés y validación móvil**: diseño de baterías de pruebas de choque (stress tests) con perturbaciones sintéticas de mercado (caídas del -10% y -20%, incrementos súbitos de correlación sectorial) junto con análisis de validación hacia adelante (Walk-Forward) para mitigar el sobreajuste de los parámetros.

A. Configuración del Sistema

A.1. Fichero principal: config.yaml

El fichero `config/config.yaml` centraliza todos los parámetros operativos del sistema. A continuación se reproduce la configuración de producción utilizada durante la validación en *paper trading*:

Listing A.1: config/config.yaml — configuración de producción

```
system:
  name: "Sistema␣Agentico␣de␣Inversion"
  version: "2.0.0"
  trading_mode: PAPER   # PAPER | LIVE

markets:
  - id: IBEX35
    exchange: BME
    open_local: "09:00"
    close_local: "17:30"
    timezone: "Europe/Madrid"
  - id: SP500
    exchange: NYSE
    open_local: "09:30"
    close_local: "16:00"
    timezone: "America/New_York"

aggregator:
  window_seconds: 300               # dedup intra-lote (5 min)
```

```
  cross_source_window_seconds: 3600 # memoria rodante multi-fuente
  multi_source_confidence_bonus: 0.05
  max_confidence_bonus: 0.15

searchers:
  cross_market:
    interval_minutes: 60
    min_move_pct: 1.0
    movers_count: 25
    sector_ttl_hours: 24
  news:
    interval_minutes: 30
    event_driven: true
  social:
    interval_minutes: 10
    apify_batch_size: 20
    apify_max_posts: 100
  premarket:
    interval_minutes: 5
    lead_minutes: 10
    top_movers: 20
  technical:
    interval_minutes: 10
    ohlcv_bars: 60
    ohlcv_interval: "5␣mins"
    batch_size: 50
  fundamental:
    interval_minutes: 30
    batch_size: 35

hitl:
  channel: telegram
  approval_ttl_seconds: 300
  buttons: ["approve", "reject", "pause_1h"]

monitoring:
  pnl_review_minutes: 60
  stop_loss_check_minutes: 15
```

```
llm:
  provider: deepseek
  reasoning_model: deepseek-v4-flash
  tool_model: deepseek-v4-flash
  temperature: 0.2
  max_tokens: 1024
  evaluation_engine: llm
  prompts_dir: prompts

advisor:
  enabled: true
  report_interval_minutes: 60
  market_hours_only: true
  chat_max_tokens: 800
  history_turns: 10
  prompt_file: "advisor/portfolio.md"
  editable_config:
    - "config/decisor_weights.yaml"
    - "config/risk_limits.yaml"
    - "config/config.yaml"

simulation:
  enabled: true
  tickers: [SAN.MC, BBVA.MC, AAPL, MSFT]
  ohlcv_bars: 200
  ohlcv_interval: "1␣day"
  backtest:
    initial_equity: 100000
    holding_bars: 20
    commission_pct: 0.001
  param_grid:
    short: [10, 20]
    long: [50, 100]
    sl_pct: [0.02, 0.03]
    rr: [2.0]

metrics:
```

```
  port: 8000
  bind_address: "0.0.0.0"
```

A.2. Fichero de límites de riesgo: risk_limits.yaml

Listing A.2: config/risk_limits.yaml

```
exposure:
  max_per_asset_pct: 10       # maximo 10% del capital por activo
  max_per_sector_pct: 25      # maximo 25% por sector
  max_total_at_risk_pct: 30   # maximo 30% de capital en riesgo
correlation:
  max_with_open_positions: 0.7
liquidity:
  min_avg_volume_20d: 100000
position_sizing:
  default_pct: 5
  min_pct: 1
  max_pct: 10
stop_loss:
  min_distance_pct: 0.5
  max_distance_pct: 5.0
take_profit:
  min_pct: 1.5
  max_pct: 3.0
risk_reward:
  min_ratio: 2.0
var:
  confidence: 0.95
  max_portfolio_var_pct: 5.0
execution:
  max_price_deviation_pct: 2.0
kill_switch:
  max_intraday_drawdown_pct: 3.0
  max_consecutive_losses: 5
```

A.3. Fichero de pesos del decisor: decisor_weights.yaml

Listing A.3: config/decisor_weights.yaml

```
weights:
  conservative:   0.30
  moderate:       0.30
  high_risk:      0.30
  sensationalist: 0.10
thresholds:
  min_final_score:     0.59   # score ponderado minimo para operar
  min_evaluator_score: 0.50   # umbral de aprobacion por evaluador
  min_consensus_count: 3      # minimo de evaluadores en acuerdo
  min_expectancy_pct:  0.30   # expectancy minima (%)
track_record:
  rebalance_every_n_trades: 50
```

A.4. Variables de entorno (.env)

El sistema requiere las siguientes variables de entorno (fichero `.env` en la raíz del proyecto):

Listing A.4: Plantilla de variables de entorno (.env.example)

```
# ---- Modo de trading ----
TRADING_MODE=PAPER          # PAPER | LIVE

# ---- IBKR ----
IBKR_HOST=127.0.0.1
IBKR_PORT=4002              # 4002=paper, 4001=live
IBKR_CLIENT_ID_READ=10
IBKR_CLIENT_ID_WRITE=11
IBKR_ACCOUNT=DU1234567

# ---- LLM (DeepSeek) ----
DEEPSEEK_API_KEY=sk-...
# Alternativas:
# OPENAI_API_KEY=sk-...
```

```
# ANTHROPIC_API_KEY=sk-ant-...
# AZURE_OPENAI_API_KEY=...
# AZURE_OPENAI_ENDPOINT=https://<recurso>.openai.azure.com
# AZURE_OPENAI_DEPLOYMENT=<nombre-deployment>
# OPENAI_API_VERSION=2024-06-01

# ---- Noticias ----
NEWSAPI_KEY=...
MARKETAUX_KEY=...
FINNHUB_KEY=...
ALPHAVANTAGE_KEY=...

# ---- Redes sociales ----
APIFY_API_TOKEN=...
RAPIDAPI_KEY=...      # alternativa: Socialgrep + Finance Sentiment

# ---- Telegram HITL ----
TELEGRAM_BOT_TOKEN=...
TELEGRAM_AUTHORIZED_USERS=123456789,987654321

# ---- Persistencia ----
POSTGRES_DSN=postgresql://agentic:password@localhost:5432/agentic
REDIS_URL=redis://localhost:6379/0
QDRANT_URL=http://localhost:6333
```

B. Métricas y Observabilidad

B.1. Métricas Prometheus completas

El sistema expone 11 métricas Prometheus en `http://0.0.0.0:8000/metrics`. La tabla B.1 las describe con detalle, indicando su funcionalidad dentro del pipeline de observabilidad.

Tabla B.1: Métricas Prometheus del sistema (detalle completo)

Métrica	Descripción
`agentic_opportunities_total`	Oportunidades detectadas acumuladas por buscador (`agent_source`).
`agentic_decisions_total`	Decisiones emitidas clasificadas por estado: `OPERATE` o `NO_OPERATE`.
`agentic_orders_total`	Órdenes enviadas al bróker según estado de ejecución (`status`).
`agentic_risk_rejections_total`	Oportunidades rechazadas determinísticamente por el Risk Filter.
`agentic_kill_switch_active`	Indicador de estado del sistema: 1 si el kill switch está activo, 0 si está inactivo.
`agentic_signal_to_order_seconds`	Histograma de latencia desde la detección de señal hasta el envío de la orden (buckets de 1s a 600s).
`agentic_pnl_unrealized_pct`	P&L no realizado de las posiciones abiertas expresado como porcentaje del equity total.
`agentic_open_positions`	Número actual de posiciones abiertas simultáneamente en el bróker.
`agentic_consecutive_losses`	Contador acumulado de pérdidas consecutivas actuales.
`agentic_missing_stop_loss_total`	Contador de posiciones abiertas detectadas por el monitor sin orden de stop-loss activa.
`agentic_monitoring_events_total`	Eventos de monitorización desglosados por tipo (`pnl_alert`, `sl_alert`, `kill_switch`) y severidad (`info`, `warning`, `critical`).

B.2. Dashboard de Grafana

El fichero `infra/grafana/dashboards/agentic_overview.json` provee un dashboard pre-configurado con los siguientes paneles:

- **Pipeline Overview**: tasa de oportunidades, decisiones y órdenes en las últimas 24 horas.

- **Risk & Safety**: kill switch status, pérdidas consecutivas, rechazos por Risk Filter.
- **Portfolio**: P&L no realizado, posiciones abiertas, posiciones sin SL.
- **Latency**: histograma de latencia señal-orden con percentiles p50, p90, p95.
- **Searchers**: desglose de oportunidades detectadas por buscador (gráfico de barras apiladas).

El dashboard se auto-provisiona al arrancar el contenedor de Grafana mediante los ficheros de provisioning en `infra/grafana/provisioning/`.

B.3. Dashboard Streamlit: agent_monitor.py

El dashboard Streamlit (`dashboard/agent_monitor.py`) complementa a Grafana con una vista orientada al operador en lugar de a la infraestructura. Se ejecuta con:

Listing B.1: Arranque del dashboard Streamlit

```
streamlit run dashboard/agent_monitor.py
```

El dashboard se conecta directamente a PostgreSQL y muestra:

- **Timeline de eventos**: todos los eventos del pipeline en orden cronológico, con conversión automática de UTC a la zona horaria local del operador.
- **Tabla de oportunidades**: ticker, buscador origen, dirección, score inicial, entrada/TP/SL, justificación del LLM.
- **Tabla de decisiones**: score final ponderado, expectancy, R/R, desglose por evaluador, razón de la decisión.
- **Tabla de órdenes**: estado de ejecución (FILLED/PARTIAL/CANCELLED), precio de llenado, P&L realizado.
- **Tabla de llamadas LLM**: modelo usado, tokens consumidos (prompt + completion), latencia, coste estimado.

B.4. Audit Logger

El `AuditLogger` escribe cada evento significativo en formato JSON estructurado al logger estándar de Python (`agentic.audit`). Cada entrada incluye:

- `timestamp_utc`: timestamp ISO 8601 en UTC.

- `event_type`: tipo de evento (`opportunity`, `evaluation`, `decision`, `hitl`, `execution`, `monitoring`).
- `payload`: datos específicos del evento (en función del tipo).

En producción, estos logs pueden dirigirse a un SIEM (Splunk, Elastic) o a un sistema de almacenamiento de auditoría inmutable (WORM) para cumplir con los requisitos de MiFID II y DORA.

Referencia bibliográfica

[A+24] A. F. Ansari et al. Chronos: Learning the language of time series, 2024.

[ABB+19] Saleema Amershi, Andrew Begel, Christian Bird, Robert DeLine, Harald Gall, Ece Kamar, Nachiappan Nagappan, Besmira Nushi, and Thomas Zimmermann. Software engineering for machine learning: A case study. In *Proceedings of the 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, pages 291–300. IEEE, 2019.

[Ald13] I. Aldridge. *High-Frequency Trading: A Practical Guide to Algorithmic Strategies and Trading Systems*. Wiley, 2nd edition, 2013.

[Ara19] D. Araci. FinBERT: Financial sentiment analysis with pre-trained language models, 2019.

[Aru23] Devika Arumugam. Algorithmic trading: Intraday profitability and trading behavior. *Economic Modelling*, 128:106521, 2023.

[BS24] Somraj Bhattacharya and Pradeep H. Sadarangani. The evolution of robo-advisory services in wealth management: A systematic literature review. *International Journal of Information Management Data Insights*, 4(1):100215, 2024.

[CC24] Rosella Carè and Douglas Cumming. Technology and automation in financial trading: A bibliometric review. *Research in International Business and Finance*, 71:102471, 2024.

[CG16] T. Chen and C. Guestrin. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794, 2016.

[DSC25] Soumyadarshan Dash, Rajiv Sharma, and Melanie Chen. Human-in-the-loop AI in financial services: Data engineering that enables judgment at scale. *Journal of Computer Science and Technology Studies*, 7(7):228–236, 2025.

[Eur22] European Parliament. Regulation (eu) 2022/2554 on digital operational resilience for the financial sector (dora). Technical report, Official Journal of the European Union, 2022.

[Eur24] European Parliament. Regulation (eu) 2024/1689 laying down harmonised rules on artificial intelligence (ai act). Technical report, Official Journal of the European Union, 2024.

[FK18] T. Fischer and C. Krauss. Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*, 270(2):654–669, 2018.

[HJM11] T. Hendershott, C. M. Jones, and A. J. Menkveld. Does algorithmic trading improve liquidity? *The Journal of Finance*, 66(1):1–33, 2011.

[Int24] International Monetary Fund. Advances in artificial intelligence: Implications for capital market activities. Chapter 3 report, IMF Global Financial Stability Report, October 2024.

[KKST17] Andrei Kirilenko, Albert S. Kyle, Mehrdad Samadi, and Tugkan Tuzun. The flash crash: High-frequency trading in an electronic market. *The Journal of Finance*, 72(3):967–998, 2017.

[L+23] Y. Li et al. TradingGPT: Multi-agent system with layered memory and distinct characters for enhanced financial trading performance, 2023.

[Lan24] LangChain AI. LangGraph: Building stateful, multi-actor applications with llms. `https://github.com/langchain-ai/langgraph`, 2024. Accedido: agosto 2026.

[LdP18] M. López de Prado. *Advances in Financial Machine Learning*. Wiley, 2018.

[LeB06] B. LeBaron. *Agent-based Financial Markets: Matching Stylized Facts with Style*. Cambridge University Press, 2006.

[Mal25] Malt. Barómetro del freelancing en españa 2025–2026, 2025. Accedido en 2026.

[MFBF+22] Silverio Martínez-Fernández, Justus Bogner, Xavier Franch, Marc Oriol, Julien Siebert, Adam Trendowicz, Anna Maria Vollmer, and Bilal Murtaza. Software engineering for AI-based systems: A survey. *ACM Transactions on Software Engineering and Methodology*, 31(2):1–59, 2022.

[Par14] Parlamento Europeo. Directiva 2014/65/ue relativa a los mercados de instrumentos financieros (MiFID II), artículo 17: Requisitos organizativos para las empresas de servicios de inversión que se dediquen a la negociación algorítmica. Diario Oficial de la Unión Europea, L 173, 2014.

[T+23] Torantulino et al. Auto-GPT: An autonomous gpt-4 experiment. `https://github.com/Significant-Gravitas/AutoGPT`, 2023. Accedido: agosto 2026.

[Tet07] P. C. Tetlock. Giving content to investor sentiment: The role of media in the stock market. *The Journal of Finance*, 62(3):1139–1168, 2007.

[W+23] S. Wu et al. BloombergGPT: A large language model for finance, 2023.

[XL25] Y. Xiao and E. Li. TradingAgents: Multi-agents llm financial trading framework, 2025.

[Y+23] S. Yao et al. ReAct: Synergizing reasoning and acting in language models, 2023.

[YLW23] H. Yang, X. Y. Liu, and C. D. Wang. FinGPT: Open-source financial large language models, 2023.

[Z+24] Y. Zhang et al. FinAgent: A multimodal foundation agent for financial trading, 2024.