



Propuesta de un modelo de negocio SaaS para la captación de usuarios investigadores e institucionales sobre temas provistos por Molecular Craft

Felipe Andrés Muñoz Vargas

felipe.munozva@usm.cl

Dr. Luis Ramírez

Profesor Guía

Resumen

Molecular Craft es una plataforma de software para modelado molecular, docking y análisis estructural en el desarrollo de fármacos. Hoy se distribuye como aplicación de escritorio: un modelo que limita su escalabilidad e impide capturar información estructurada sobre el uso de la herramienta. En el marco de su evolución hacia un modelo de Software como Servicio (SaaS), este trabajo diseña e implementa un mecanismo concreto para captar y retener una masa crítica de usuarios científicos e institucionales. Ese mecanismo es un asistente agéntico de bioinformática, integrado en una versión web de la búsqueda en el Protein Data Bank. El asistente se construye sobre un grafo de estados que orquesta nodos especializados. Resuelve proteínas y recupera literatura real desde PubMed mediante el Model Context Protocol, vinculando cada publicación a su fuente para reducir alucinaciones. Su aporte central es que, al mediar la interacción, convierte cada consulta en datos estructurados: proteínas consultadas, literatura recuperada y rutas de seguimiento. Esos datos son la materia prima para reconocer patrones de uso, entrenar modelos e impulsar la adopción. Se trata de un estudio aplicado de alcance exploratorio. Su resultado es un asistente funcional, que evidencia cómo la asistencia inteligente puede aportar valor desde el primer uso y proponerse como palanca de captación y retención hacia una futura masa crítica de usuarios.

Palabras clave: SaaS, software científico, docking molecular, efectos de red, masa crítica, agentes de IA, captura de datos estructurados, Model Context Protocol, PubMed



CONSTANCIA DE VALIDACIÓN Y CONFIDENCIALIDAD DE MONOGRAFÍA A REPOSITORIO ACADÉMICO

1.- IDENTIFICACIÓN DEL TRABAJO ACADÉMICO

Tipo de monografía (marcar una opción): ☒ Memoria o trabajo de título ☐ Tesis de Postgrado

Título del trabajo: Propuesta de un modelo de negocio SaaS para la captación de usuarios investigadores e institucionales sobre temas provistos por Molecular Craft

Nombre del candidato(a): Felipe Andrés Muñoz Vargas

Carrera / Grado: Ingeniería Civil Informática

Campus: Casa Central Valparaíso **Departamento:** Informática

2.- VALIDACIÓN DEL PROFESOR GUÍA/DIRECTOR DE TESIS

Yo, Luis Alejandro Ramírez Donoso, en mi calidad de profesor(a) guía/director(a) del trabajo académico mencionado anteriormente **DEJO CONSTANCIA** que:

- He revisado esta versión del documento y corresponde a la versión final aprobada del trabajo.
- El trabajo cumple con los requisitos académicos y de formato establecidos por la institución.

3.- EVALUACIÓN DE CONFIDENCIALIDAD POR PROPIEDAD INDUSTRIAL (marcar una opción)

☒ El trabajo **NO contiene** información que amerite confidencialidad y puede ser publicado de inmediato en repositorio con acceso abierto.

☐ El trabajo **CONTIENE** información con potenciales implicancias de propiedad industrial o intelectual y requiere un periodo de confidencialidad (**embargo**) por (**marcar una opción**):

☐ 6 meses ☐ 12 meses ☐ 2 años ☐ 3 años ☐ 5 años ☐ 10 años

Fundamentación de la necesidad de confidencialidad (obligatorio si se solicita embargo):

4.- FIRMAS

Profesor(a) guía o director(a) de memoria o tesis:

Fecha: 13-07-2026 **Firma:** 

Estudiante o Candidato(a):

Fecha: 13-07-2026 **Firma:** 

Este formulario debe ser insertado como página 2 de la memoria o tesis, completado y firmado por estudiante y profesor(a) antes de la entrega en portal PRISMA de Biblioteca USM.



Índice

3

1. Introducción	6
1.1. Descripción del problema	6
1.2. Justificación del enfoque	8
1.3. Estructura del informe	9
1.4. Objetivos	9
2. Marco Teórico	10
2.1. Masa crítica, efectos de red y adopción en plataformas digitales	10
2.1.1. Efectos de red y masa crítica	10
2.1.2. Casos de referencia en plataformas científicas SaaS	10
2.1.3. Adopción y continuidad de uso en plataformas SaaS	11
2.1.4. Congestión y el compromiso entre crecimiento y calidad	12
2.2. Agentes de IA y arquitecturas agénticas	12
2.2.1. De agentes reactivos a agentes con planificación	12
2.2.2. Patrones de diseño de sistemas agénticos	13
2.2.3. Orquestación por grafos de estados	17
2.3. Model Context Protocol (MCP) e integración con herramientas externas . .	17
2.4. Fundamentación en fuentes científicas verificables	17
2.5. Síntesis	18
3. Metodología	19
3.1. Enfoque general	19
3.2. Alcance y delimitación	19
3.3. Herramientas y tecnologías	20
3.4. Fases de trabajo	22
3.4.1. Fase 1: Migración de la búsqueda PDB a un entorno web	23
3.4.2. Fase 2: Diseño e implementación de la arquitectura agéntica	23
3.4.3. Fase 3: Integración de fuentes científicas externas	24
3.4.4. Fase 4: Implementación del comportamiento proactivo	24
3.4.5. Fase 5: Demostración como mecanismo de captación	24
3.5. Criterios de validación	24
3.6. Diseño de la validación futura	25
4. Desarrollo	27
4.1. Arquitectura general de la solución	27
4.2. Migración de la búsqueda en el Protein Data Bank	28
4.2.1. Modos de búsqueda y parámetros	28
4.2.2. Presentación de resultados y acciones	29
4.3. Backend agéntico	29
4.3.1. Organización del proyecto	29
4.3.2. Grafo de estados y nodos	30



4.3.3. Memoria conversacional persistente	4
4.3.4. API y contrato de <i>streaming</i>	31
4.4. Integración con fuentes científicas externas	32
4.5. Comportamiento proactivo del asistente	33
4.6. Pruebas	34
4.7. Limitaciones técnicas del prototipo	36
4.8. Demostración: comparación escritorio <i>vs.</i> web asistida	36
4.9. De la solución técnica al resultado estratégico: cadena de valor	39
4.10. Beneficios por tipo de stakeholder	39
4.11. Discusión de resultados	41
4.11.1. Lectura frente a los objetivos	41
4.11.2. Contraste con el estado del arte	41
4.11.3. Validez externa y alcance de la generalización	42
4.11.4. Efecto de las limitaciones sobre lo que puede afirmarse	42
5. Conclusiones	43
Anexos	45
Anexo A: Capturas del flujo del asistente	45
Referencias	47

Índice de figuras

1. Bloque de construcción: el LLM aumentado	14
2. Patrón de encadenamiento de prompts (<i>prompt chaining</i>)	15
3. Patrón de enrutamiento (<i>routing</i>)	15
4. Patrón de paralelización (<i>parallelization</i>)	15
5. Patrón orquestador-trabajadores (<i>orchestrator-workers</i>)	16
6. Patrón evaluador-optimizador (<i>evaluator-optimizer</i>)	16
7. Arquitectura general de la solución.	28
8. Interfaz web de búsqueda en el Protein Data Bank	29
9. Grafo de estados del asistente y sus nodos.	31
10. Flujo de búsqueda en la versión de escritorio.	37
11. Flujo de búsqueda en la versión web asistida.	38
12. Selección de proteína y resumen del asistente	45
13. Tabla de publicaciones y sugerencias proactivas	46

Índice de cuadros

1. Comparación de marcos de orquestación agéntica	21
2. Comparación de modelos de lenguaje	22



3.	Correspondencia entre las fases de trabajo y los objetivos específicos.	5
4.	Métricas e instrumentos propuestos para la validación futura	23
5.	Principales <i>endpoints</i> expuestos por el backend.	26
6.	Grupos de pruebas automatizadas del <i>backend</i> del asistente.	32
7.	Cobertura de código por componente (<i>pytest-cov</i>).	34
8.	Criterios de aceptación y alcance de lo validado	35
9.	Beneficios de la solución por tipo de <i>stakeholder</i>	40



1. Introducción

Antes de introducir el tema principal, conviene una breve presentación del proyecto de referencia. Molecular Craft es una herramienta destinada a científicos e investigadores químicos que trabajan en el descubrimiento de fármacos y en el desarrollo químico. Nace de una necesidad concreta: contar con un software único que integre aplicaciones y funcionalidades hoy dispersas en distintas plataformas y sistemas operativos, y sujetas además a problemas de versionamiento.

Se trata de un software impulsado por inteligencia artificial, que integra varias tecnologías del diseño de fármacos y la investigación química: Autodock, Autodock Vina, Autodocktools, ChEMBL, Protein Data Bank y generadores de ligandos, entre otras (Molecular Craft, 2025). Con esta aplicación, el usuario selecciona las biomoléculas a probar y elige la proteína a estudiar. Puede entonces visualizar la proteína en 3D, posicionar las biomoléculas sobre ella y observar su interacción. Obtiene, asimismo, indicadores como la energía de enlace, los niveles de energía, los puentes de hidrógeno y los niveles de toxicidad.

1.1. Descripción del problema

El actual modelo de distribución de Molecular Craft se basa en la instalación local. Esta modalidad limita su escalabilidad, dificulta la entrega de funcionalidades diferenciadas y restringe la capacidad de construir relaciones sostenibles con los usuarios. Impide, en particular, generar un sistema de suscripción que permita identificar, atraer y fidelizar perfiles científicos calificados, cuyo conocimiento especializado constituye un activo estratégico para la plataforma. A ello se suma la imposibilidad de capturar información estructurada sobre esos usuarios, lo que impide reconocer patrones de uso, intereses específicos o áreas de investigación dominantes.

Estas limitaciones no son exclusivas de Molecular Craft. La literatura sobre software científico documenta que la fricción de instalación y la coordinación de dependencias figuran entre las principales barreras de entrada de las herramientas bioinformáticas. Documenta, además, que las herramientas diseñadas sin considerar la usabilidad tienden a no sostener su uso en el tiempo (Mangul et al., 2019). Un estudio reciente lo confirma sobre el terreno. Basado en una revisión de la literatura y en entrevistas a investigadores noveles en genómica, identifica los factores que degradan la adopción y el uso sostenido de estas herramientas. Entre ellos, la dificultad de instalación y despliegue, junto con la documentación deficiente (Patel et al., 2026). La evidencia externa respalda, por tanto, el diagnóstico estructural que motiva la evolución de Molecular Craft hacia un modelo accesible vía navegador.

Conviene precisar desde el inicio que este problema opera en dos niveles complementarios. El primero es un problema estructural directamente observable: el modelo de instalación local limita la escalabilidad, dificulta la actualización centralizada, restringe la entrega de funcionalidades diferenciadas e impide capturar información estructurada sobre el uso. El segundo es un problema de negocio todavía no cuantificado con evidencia interna. El efecto preciso de esas restricciones sobre la captación, la retención y la continuidad de uso no ha sido medido: la plataforma aún carece de la instrumentación necesaria para hacerlo. En



consecuencia, este trabajo se orienta en primer término a cerrar esa brecha de instrumentación y a fortalecer la propuesta de valor del producto, antes que a demostrar impactos comerciales finales.

El avance de las soluciones científicas basadas en SaaS ha demostrado que existe una alternativa viable. Hoy es posible ofrecer herramientas complejas de modelado molecular, acoplamiento molecular (docking) y diseño de fármacos asistido por computador (CADD) a través de plataformas accesibles vía navegador, bajo esquemas de suscripción escalables y controlables. La trayectoria de estas soluciones ofrece un antecedente útil. Ya en la década pasada, plataformas como e-Science Central mostraron la viabilidad de ejecutar aplicaciones científicas en la nube (Watson et al., 2013). El sector documentaba entonces cómo las soluciones SaaS validadas permiten cumplir normativas regulatorias estrictas, a la vez que optimizan los costos operacionales y la gestión de versiones (Clifford, 2018). Esos trabajos se citan aquí como antecedentes tempranos del modelo, no como estado del arte actual.

La evidencia reciente y directamente pertinente al dominio la aporta AIDDISON, que provee herramientas de IA y CADD para descubrimiento de fármacos en una plataforma SaaS segura y accesible vía navegador (Rusinko et al., 2024). La implementación de flujos de trabajo científicos mediante tecnologías cloud-native como Apache Airflow y Kubernetes, aplicadas al docking molecular, refuerza la viabilidad técnica de este tipo de arquitectura (Medeiros et al., 2023). Una tendencia creciente muestra, por último, cómo las empresas de IA en descubrimiento de fármacos abren sus plataformas mediante esquemas SaaS por suscripción, democratizando el acceso a tecnología de punta (Pratap, 2024).

La literatura arbitrada sobre adopción de SaaS permite precisar por qué este modelo se ajusta al perfil de usuario buscado. Un estudio sobre los determinantes de la adopción, realizado desde el marco tecnológico-organizacional-ambiental, halla que el modelo resulta más atractivo allí donde la capacidad técnica interna de la organización es menor (Shapouri et al., 2024). Es exactamente la situación de un laboratorio o grupo de investigación, que rara vez cuenta con soporte informático dedicado. Sobre ese perfil recae, por tanto, todo el costo de instalar y mantener una aplicación de escritorio.

Conviene, sin embargo, no confundir dos decisiones distintas. Adoptar una plataforma es una cosa; permanecer en ella, otra. La evidencia sobre productos SaaS por suscripción es clara al respecto. La continuidad de uso no depende de la mera disponibilidad del servicio, sino de la satisfacción y del valor percibido. Y ambos se sostienen en el ajuste entre la tarea del usuario y la tecnología que se le ofrece (Cao et al., 2026). De ello se sigue el supuesto que articula este trabajo: migrar a la nube habilita la captación, pero no garantiza la permanencia. Esta última exige entregar valor en el uso, y es ahí donde se sitúa el mecanismo que aquí se propone.

La magnitud de esta oportunidad puede dimensionarse a partir del mercado de la inteligencia artificial aplicada al descubrimiento de fármacos. Distintas consultoras lo estiman entre USD 2,4 y 6,9 mil millones en 2025, con un crecimiento anual compuesto de dos dígitos proyectado hacia la próxima década (Grand View Research, 2025; Precedence Research, 2025). Dentro de ese mercado, el segmento de software —y en particular las soluciones en la nube— concentra la mayor parte del valor. Las instituciones académicas y de investigación constituyen, a su vez, el segmento de más rápido crecimiento (Grand View Research, 2025): precisamente



el perfil de usuario que Molecular Craft busca captar. Frente a ese potencial, el modelo de escritorio actual carece de las capacidades de colaboración, actualización centralizada y accesibilidad inmediata propias de una plataforma en la nube. Ello limita estructuralmente la capacidad de captar y retener a ese perfil de usuarios, y motiva la evolución hacia un modelo SaaS.

En este contexto, el presente trabajo se enmarca en el potencial de un modelo SaaS para Molecular Craft, orientado a captar una masa crítica de usuarios científicos e institucionales. Se concentra, dentro de ese marco, en un mecanismo concreto para lograrlo: el diseño e implementación de un asistente agéntico de bioinformática. Apoyado en modelos de lenguaje y en la integración de herramientas externas, el asistente acompaña al usuario en la consulta de proteínas, la búsqueda bibliográfica y la conversación con memoria. Conviene anticipar, no obstante, el alcance de esta contribución. El trabajo no demuestra empíricamente un aumento efectivo de la captación o la retención de usuarios. Valida, en cambio, la factibilidad técnica y el valor funcional de un mecanismo que podría contribuir a dichos resultados en operación real, siempre que se complemente con instrumentación analítica y evaluación con usuarios.

Al mediar la interacción con la plataforma, el asistente ataca además una de las carencias ya señaladas del modelo de escritorio. Cada consulta se convierte en datos estructurados: proteínas consultadas, literatura recuperada y rutas de seguimiento aceptadas. Esos datos son la materia prima para reconocer patrones de uso e informar los mecanismos de captación y retención que persigue el trabajo.

La transición hacia un esquema SaaS responde, así, tanto a una oportunidad de mercado como a la necesidad de superar limitaciones concretas de adopción, actualización, trazabilidad y escalabilidad. Conviene, con todo, precisar el alcance del diagnóstico en su segundo nivel. Los indicadores que permitirían dimensionar internamente el problema no están hoy disponibles. Faltan el número de usuarios activos, la frecuencia de uso por módulo, la tasa de retención y la conversión esperada. Falta también evidencia sobre la fricción de instalación y sobre el abandono asociado al modelo de escritorio. Su dimensionamiento se sustenta, por tanto, en la evidencia de mercado y en el análisis estructural de las limitaciones del modelo local. Dotar a la plataforma de esa capacidad de medición es, precisamente, uno de los aportes del asistente propuesto y una de las líneas de trabajo futuro definidas en el Capítulo 5.

1.2. Justificación del enfoque

Abordar la captación y retención de usuarios en la transición hacia un modelo SaaS admite distintas palancas. Entre ellas figuran los ajustes en el modelo de precios y suscripción, las mejoras en la incorporación (*onboarding*) de nuevos usuarios y la apertura de una API pública. Figuran también las acciones de difusión y el fortalecimiento del valor entregado dentro del propio producto mediante asistencia inteligente. Este trabajo se concentra deliberadamente en esta última palanca —un asistente agéntico de bioinformática— por tres razones.

La primera es su alineamiento con el marco de efectos de red y masa crítica que enmarca el problema (Capítulo 2). A diferencia de las palancas de precio o difusión, un asistente que aporta valor desde el primer uso incrementa el valor percibido por cada usuario. Con ello contribuye a acelerar la llegada al umbral de masa crítica.



La segunda es su pertinencia con el perfil de usuario objetivo. El segmento científico e institucional valora la rigurosidad y la trazabilidad de la información. Un asistente que resuelve proteínas y fundamenta sus respuestas en literatura verificable ataca, por tanto, el núcleo de la propuesta de valor de la plataforma.

La tercera es su carácter de artefacto concreto, evaluable y replicable. Intervenciones como una estrategia de precios resultan difíciles de aislar y medir en un piloto. El asistente, en cambio, es un componente de software acotado: puede construirse, demostrarse sobre una sección representativa y, por diseño, extenderse al resto de los módulos.

1.3. Estructura del informe

El informe se organiza como sigue. El presente capítulo introduce el proyecto, el problema abordado y los objetivos. El Capítulo 2 presenta el marco teórico, articulado en cuatro ejes. El primero cubre los efectos de red, la masa crítica y la adopción y continuidad de uso en plataformas SaaS. Los tres restantes fundamentan la dimensión técnica: las arquitecturas agénticas basadas en modelos de lenguaje, el Model Context Protocol (MCP) para la integración con herramientas externas y la fundamentación de las respuestas en fuentes científicas verificables. El Capítulo 3 describe la metodología de trabajo. El Capítulo 4 reporta el desarrollo, los resultados y su discusión. El Capítulo 5, finalmente, presenta las conclusiones y las líneas de trabajo futuro.

1.4. Objetivos

El objetivo general es diseñar e implementar un asistente agéntico de bioinformática para Molecular Craft, integrado en una versión web de la plataforma. Sobre ese artefacto se evalúan dos aspectos: su viabilidad técnica y su potencial como mecanismo de captación y retención de una masa crítica de usuarios científicos e institucionales. Todo ello, en el marco de la evolución de la herramienta hacia un modelo de plataforma en la nube (SaaS).

Objetivos específicos:

- Migrar a un entorno web una sección de Molecular Craft —la búsqueda en el Protein Data Bank—, como base para integrar asistencia inteligente.
- Diseñar e implementar la arquitectura agéntica del asistente, basada en un grafo de estados con memoria conversacional persistente.
- Integrar fuentes científicas externas —la API del Protein Data Bank y un servidor MCP de PubMed— para dotar de rigurosidad y trazabilidad a las respuestas del asistente.
- Dotar al asistente de un comportamiento proactivo que recupere literatura científica, vincule las publicaciones a la conversación y proponga nuevas líneas de búsqueda.
- Mostrar, a través del propio asistente implementado en Molecular Craft web, cómo la asistencia agéntica puede operar como mecanismo de captación y *engagement* de usuarios.



2. Marco Teórico

Para fundamentar el diseño de un asistente agéntico de bioinformática para Molecular Craft —contribución central de este trabajo— y su rol como mecanismo de captación de usuarios, este capítulo articula cuatro ejes conceptuales. El primero aborda los efectos de red, la masa crítica y los mecanismos de adopción y retención de usuarios en plataformas digitales, que enmarcan la dimensión de negocio. Los tres restantes fundamentan la dimensión técnica del asistente. El segundo aborda las arquitecturas agénticas basadas en modelos de lenguaje. El tercero, el Model Context Protocol (MCP) como interfaz de integración con herramientas externas. El cuarto, la fundamentación de las respuestas en fuentes científicas verificables, como mecanismo para reducir alucinaciones.

2.1. Masa crítica, efectos de red y adopción en plataformas digitales

2.1.1. Efectos de red y masa crítica

Los efectos de red describen el fenómeno por el cual el valor que un producto o servicio entrega a cada usuario aumenta a medida que crece el número total de usuarios. En plataformas digitales, estos efectos constituyen tanto un motor catalizador del crecimiento del ecosistema como una barrera de entrada frente a competidores, dado que un sistema con una base amplia y activa resulta difícil de desplazar (Cervini & Joglekar, 2025). El concepto de masa crítica se refiere al umbral mínimo de usuarios a partir del cual estos efectos se vuelven autosostenibles. Por debajo de ese umbral, la plataforma lucha por retener participantes. Por encima, el crecimiento tiende a reforzarse a sí mismo.

Para una herramienta científica como Molecular Craft, alcanzar una masa crítica de investigadores e instituciones es estratégicamente relevante. Habilita dinámicas de colaboración, intercambio de datos y validación entre pares que un modelo de escritorio aislado no puede ofrecer. Las ventajas de red condicionan, además, tanto las estrategias de captación como las de retención. Dotar a la plataforma de un mecanismo que aporte valor desde el primer uso —como una asistencia inteligente— resulta, por tanto, clave para acelerar la llegada a ese umbral.

2.1.2. Casos de referencia en plataformas científicas SaaS

La viabilidad de este modelo en el ámbito científico se observa en plataformas SaaS ya consolidadas. Sus cifras ilustran dos cosas: la escala alcanzable y la retención característica de un sistema con fuertes efectos de red.

El primer caso es Schrödinger, plataforma de descubrimiento de fármacos basada en computación física. En el año fiscal 2024 reportó ingresos por software cercanos a los USD 180 millones, con un crecimiento en torno al 13 % respecto de 2023. Entre los clientes con un valor de contrato anual igual o superior a USD 500.000, su tasa de retención fue del 100 % (Schrödinger, Inc., 2025). El segundo es Benchling, plataforma en la nube para investigación y desarrollo en ciencias de la vida. Hacia mediados de 2024 alcanzó unos USD 210 millones



de ingresos recurrentes anuales (ARR), con cerca de 1.200 clientes y una retención neta por ingresos superior al 150 % (Asplund, 2024).

Ambos casos son relevantes para Molecular Craft en dos sentidos. Primero, confirman la existencia de un mercado dispuesto a adoptar herramientas científicas complejas bajo un modelo de suscripción en la nube. Segundo, sus altas tasas de retención evidencian la lógica autosostenible de la masa crítica. Una vez que una base de investigadores e instituciones integra la plataforma en su flujo de trabajo, los costos de cambio y las dinámicas de colaboración refuerzan la permanencia. De hecho, Benchling atribuye parte de su crecimiento a los efectos de red y a los flujos de datos: una mayor cantidad de científicos en la plataforma enriquece el propio producto (Asplund, 2024). Es, precisamente, el tipo de dinámica que un mecanismo de valor desde el primer uso —como la asistencia inteligente propuesta— busca acelerar.

Conviene, con todo, precisar el estatuto de esta evidencia. Ambas cifras provienen de reportes financieros y de análisis de mercado, no de estudios arbitrados. Sirven para dimensionar el fenómeno; no para explicarlo. La pregunta de por qué una organización adopta un modelo SaaS —y, sobre todo, de por qué permanece en él— exige acudir a la literatura académica sobre adopción y continuidad de uso, que se revisa a continuación.

2.1.3. Adopción y continuidad de uso en plataformas SaaS

La literatura arbitrada sobre sistemas de información distingue dos decisiones que conviene no confundir. Una es la decisión inicial de **adoptar** una plataforma. Otra, posterior, es la decisión de **seguir usándola**. Cada una responde a factores distintos. La distinción importa aquí porque el objetivo general de este trabajo compromete ambas: captación y retención.

Sobre la primera, un estudio reciente examina los determinantes de la adopción de SaaS desde el marco tecnológico-organizacional-ambiental (TOE), sobre más de un millón de observaciones de empresas a lo largo de nueve años (Shapouri et al., 2024). Dos de sus hallazgos son directamente pertinentes. El primero: el tamaño del departamento de TI incide *negativamente* en la adopción. Es decir, el modelo SaaS resulta más atractivo allí donde la capacidad técnica interna es menor, pues sustituye aquello que la organización no puede sostener por sí misma. El segundo: la adopción por parte de organizaciones pares la impulsa de forma positiva. Ambos hallazgos refuerzan el diagnóstico de este trabajo. Un laboratorio o grupo de investigación rara vez dispone de soporte informático dedicado, de modo que la fricción de instalación del modelo de escritorio recae directamente sobre el investigador. Y la segunda dinámica —la imitación entre pares— es, en el fondo, un efecto de red por la vía normativa.

Sobre la segunda decisión, la evidencia es aún más relevante para la tesis central. Un estudio arbitrado sobre productos SaaS por suscripción modela la continuidad de uso combinando el modelo de expectativa-confirmación con el de ajuste entre tarea y tecnología (Cao et al., 2026). Sus resultados muestran que el ajuste tarea-tecnología y la imagen de capacidad del proveedor inciden sobre la satisfacción y el valor percibido. Y son esos dos factores —no la mera disponibilidad del servicio— los que determinan la continuidad de uso.

De ahí se sigue una consecuencia que ordena todo el trabajo: **migrar a la nube no basta**. La migración habilita la adopción, pero la permanencia depende del valor que el



usuario percibe al usar la herramienta. Es exactamente el punto donde interviene el asistente propuesto. El asistente resuelve la tarea concreta del investigador: identificar una proteína y su literatura asociada, sin salir de la plataforma. Opera así sobre las dos variables que la literatura señala como determinantes. Mejora el ajuste entre la tarea y la tecnología, y eleva el valor percibido desde la primera interacción.

Corresponde, no obstante, delimitar el alcance de esta fundamentación. La literatura citada sustenta el *mecanismo* sobre el que actúa el asistente; no demuestra su efecto en este caso particular. Este trabajo no mide satisfacción, valor percibido ni continuidad de uso. La cadena que conecta la asistencia inteligente con la retención queda, por tanto, teóricamente fundada y empíricamente pendiente, tal como se explicita en la Sección 3.6.

2.1.4. Congestión y el compromiso entre crecimiento y calidad

El crecimiento de la base de usuarios no es, sin embargo, un proceso libre de tensiones. Modelos formales de competencia entre plataformas muestran que la congestión puede degradar la calidad de los servicios ofrecidos, especialmente cuando fuertes efectos de red atraen a muchos usuarios simultáneamente (Katsamakos & Sánchez-Cartas, 2025). Este compromiso entre crecimiento y calidad es particularmente pertinente en una plataforma científica en la nube, donde el desempeño de cargas computacionalmente intensivas —como el docking molecular— debe escalar sin deteriorar la experiencia. Toda estrategia de captación —incluida la asistencia inteligente que propone este trabajo— debe, por tanto, equilibrar la incorporación de nuevos usuarios con la preservación de la calidad de servicio.

Un enfoque más reciente amplía esta idea mediante la noción de *efectos de red generados por máquinas*. Cuando una plataforma incorpora inteligencia artificial, los datos que los usuarios generan al utilizarla permiten entrenar mejores modelos. Esos modelos mejoran el servicio, lo que a su vez atrae y retiene a más usuarios. Se forma así un ciclo de retroalimentación —un círculo virtuoso de datos— que refuerza los efectos de red (Cervini & Joglekar, 2025). En un Molecular Craft enriquecido con un asistente de IA, esta dinámica es directa. Cada interacción del usuario produce datos estructurados sobre su uso. Gestionados adecuadamente, esos datos permiten mejorar el asistente y la plataforma, y refuerzan los efectos de red que sostienen la masa crítica.

2.2. Agentes de IA y arquitecturas agénticas

2.2.1. De agentes reactivos a agentes con planificación

Un agente de IA es un sistema capaz de percibir su entorno y actuar sobre él para alcanzar objetivos. Una distinción fundamental separa a los agentes reactivos, que responden de forma inmediata a estímulos sin un modelo explícito del futuro, de los agentes proactivos o anticipatorios, que planifican secuencias de acciones orientadas a metas. Con la llegada de los modelos de lenguaje de gran escala (LLM), las arquitecturas agénticas han ganado capacidad para descomponer problemas complejos y planificar de forma multi-paso.

En esta línea se ha propuesto una arquitectura de planificación agéntica modular, formalizada desde una perspectiva neuro-inspirada (Webb et al., 2025). En ella, módulos especializados

basados en LLM descomponen la planificación en componentes que interactúan entre sí: el monitoreo de conflictos, la predicción de estados y la evaluación. La modularidad resulta clave para el asistente de bioinformática propuesto, pues permite separar responsabilidades —consulta de proteínas, búsqueda bibliográfica, agregación de resultados— en componentes especializados y coordinados.

2.2.2. Patrones de diseño de sistemas agénticos

Un referente ampliamente adoptado para el diseño de sistemas agénticos con LLM es la sistematización propuesta por Anthropic (Anthropic, 2024). Esta distingue dos categorías. Los *flujos de trabajo* (*workflows*) son sistemas en los que los LLM y las herramientas se orquestan mediante rutas de código predefinidas. Los *agentes* propiamente dichos, en cambio, dirigen dinámicamente sus propios procesos y el uso de herramientas, manteniendo el control sobre cómo alcanzan sus objetivos. El bloque de construcción básico de ambas categorías es el *LLM aumentado*: un modelo de lenguaje dotado de recuperación de información, uso de herramientas y memoria. La propia sistematización señala al MCP como una vía para implementar esa integración, lo que conecta directamente con el Eje 3 de este marco.

Sobre esa base, se identifican cinco patrones composables de flujo de trabajo (Anthropic, 2024):

- **Encadenamiento de prompts** (*prompt chaining*, Figura 2): descompone una tarea en una secuencia de pasos, donde cada llamada al LLM procesa la salida de la anterior, con posibles validaciones programáticas entre pasos.
- **Enrutamiento** (*routing*, Figura 3): clasifica una entrada y la deriva a una tarea o manejador especializado, evitando que la optimización para un tipo de entrada degrade el desempeño en los demás.
- **Paralelización** (*parallelization*, Figura 4): ejecuta varias llamadas al LLM de forma simultánea y agrega sus salidas. Admite dos variantes: el seccionamiento, que divide la tarea en subtareas independientes ejecutadas en paralelo; y la votación, que repite la misma tarea para obtener resultados diversos y mayor confianza.
- **Orquestador-trabajadores** (*orchestrator-workers*, Figura 5): un LLM central descompone dinámicamente la tarea, delega subtareas en LLM trabajadores y sintetiza sus resultados; a diferencia de la paralelización, las subtareas no están predefinidas, sino que se determinan según la entrada concreta.
- **Evaluador-optimizador** (*evaluator-optimizer*, Figura 6): un LLM genera una respuesta mientras otro la evalúa y retroalimenta en un ciclo iterativo de refinamiento.

El bloque de LLM aumentado se esquematiza en la Figura 1, y cada patrón en la figura indicada junto a su descripción.

Aunque esta taxonomía de patrones fue sistematizada por Anthropic desde la práctica industrial, es coherente con la literatura académica reciente sobre arquitecturas de agentes

basados en LLM. Diversos estudios de revisión descomponen los agentes en módulos de perfil, memoria, planificación y acción (Wang et al., 2024). Otros proponen un esquema análogo de cerebro-percepción-acción (Xi et al., 2025). Ambos marcos fundamentan la noción de LLM aumentado y sus ciclos de planificación y uso de herramientas.

La convergencia se extiende a los demás ejes. Las revisiones sobre aprendizaje de herramientas (*tool learning*) organizan el flujo en fases de planificación de la tarea, selección, invocación y generación de respuesta (Qu et al., 2025), en línea directa con el Eje 3 de este marco. Las revisiones de arquitecturas emergentes documentan de forma explícita las estructuras de orquestador-trabajador, la ejecución paralela de herramientas y los ciclos de refinamiento por retroalimentación (Masterman et al., 2024). Los estudios sobre sistemas multiagente, por su parte, formalizan las topologías de coordinación centralizada y descentralizada que subyacen a los patrones de orquestación y paralelización (Guo et al., 2024). Esta coincidencia entre la sistematización de Anthropic y la literatura revisada por pares refuerza la solidez del andamiaje conceptual sobre el que se diseña el asistente.

El asistente diseñado para Molecular Craft se sitúa en este marco como un LLM aumentado que combina dos de estos patrones. El primero es la paralelización, en su variante de seccionamiento: un *fan-out* ejecuta en paralelo la consulta de proteínas y la búsqueda bibliográfica en PubMed, y un componente dedicado agrega sus salidas. El segundo es el enrutamiento, que decide de forma condicional si reutiliza las publicaciones ya recuperadas o vuelve a consultar PubMed. Estas subtarefas están predefinidas, rasgo que distingue la paralelización del patrón orquestador-trabajadores. Su coordinación, no obstante, no se resuelve mediante rutas de código rígidas, sino sobre un grafo de estados explícito, enfoque que se detalla en el siguiente apartado. El resultado combina dos propiedades: la estructura y la trazabilidad propias de un flujo de trabajo, y un enrutamiento condicional que ajusta la invocación de herramientas según el tipo de consulta.

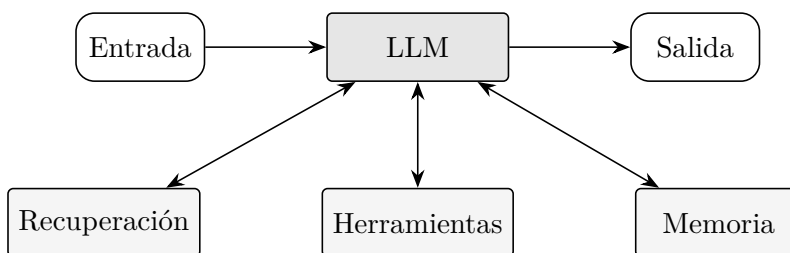


Figura 1: Bloque de construcción de los sistemas agénticos: el LLM aumentado, dotado de recuperación de información, herramientas y memoria. Elaboración propia a partir de (Anthropic, 2024).

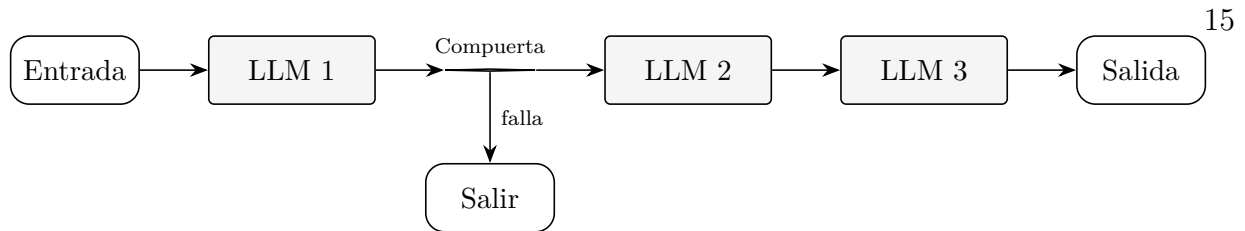


Figura 2: Patrón de encadenamiento de prompts (*prompt chaining*): cada llamada procesa la salida de la anterior, con una compuerta de validación entre pasos. Elaboración propia a partir de (Anthropic, 2024).

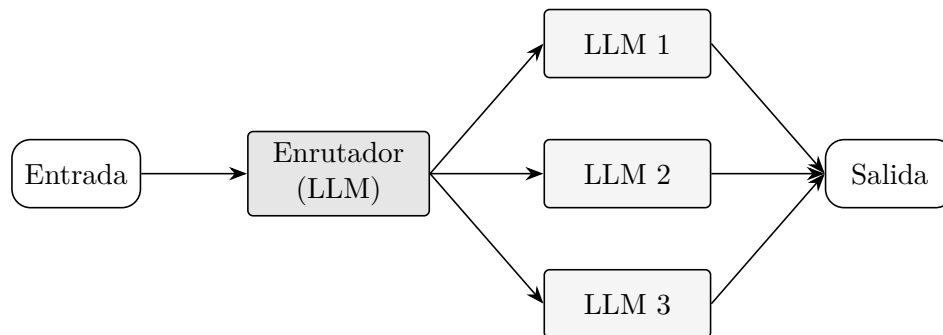


Figura 3: Patrón de enrutamiento (*routing*): un LLM clasifica la entrada y la deriva a un manejador especializado. En el asistente propuesto, un enrutamiento condicional determinista decide si se reutilizan las publicaciones en memoria o se vuelve a consultar PubMed. Elaboración propia a partir de (Anthropic, 2024).

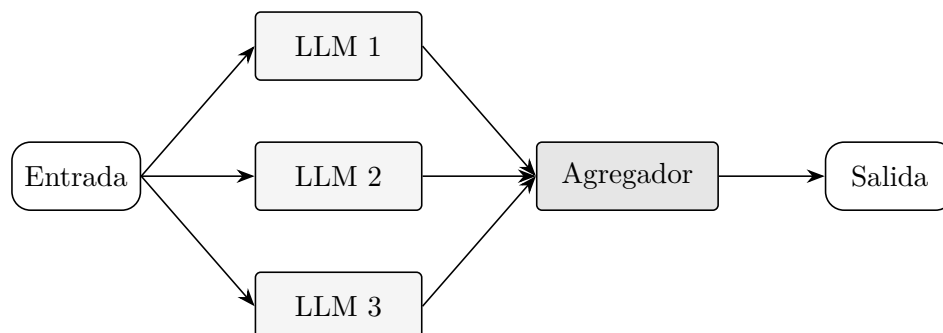


Figura 4: Patrón de paralelización (*parallelization*): varias llamadas se ejecutan en paralelo y sus salidas se agregan. A diferencia del patrón orquestador-trabajadores, las subtarefas están predefinidas. Es, junto con el enrutamiento, el patrón que sigue el asistente propuesto. Elaboración propia a partir de (Anthropic, 2024).

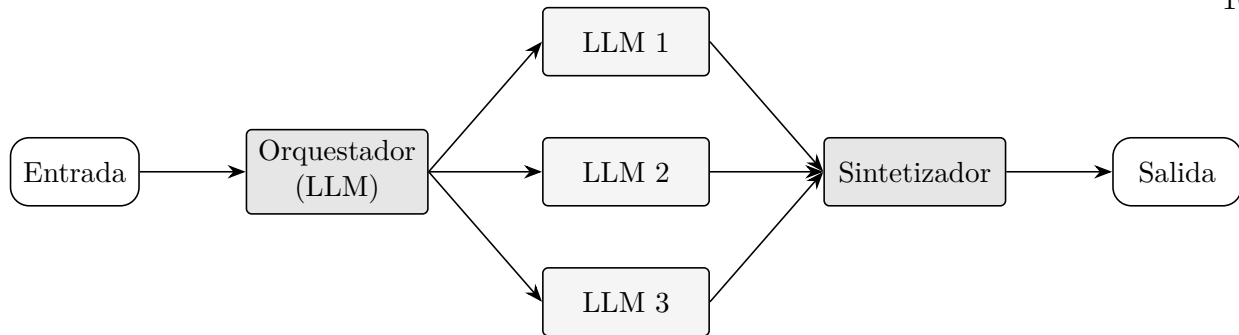


Figura 5: Patrón orquestador-trabajadores (*orchestrator-workers*): un LLM central descompone dinámicamente la tarea, delega subtarefas y sintetiza los resultados. Elaboración propia a partir de (Anthropic, 2024).

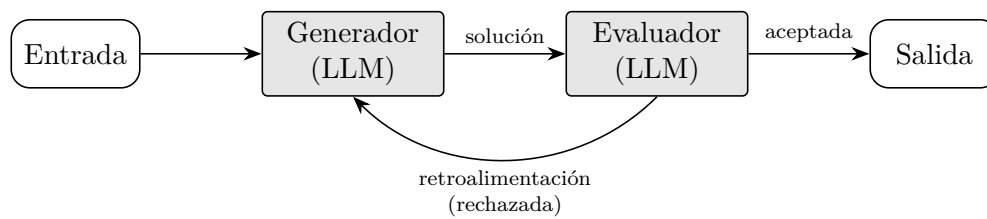


Figura 6: Patrón evaluador-optimizador (*evaluator-optimizer*): un LLM genera una respuesta y otro la evalúa y retroalimenta en un ciclo iterativo. Elaboración propia a partir de (Anthropic, 2024).



2.2.3. Orquestación por grafos de estados

Una forma efectiva de coordinar estos módulos es la orquestación mediante grafos de estados, paradigma que extiende los flujos de trabajo basados en grafos a los agentes LLM, habilitando planificación y ejecución estructuradas sobre grafos de estados explícitos (Liu et al., 2025). Este enfoque es el que adopta el asistente diseñado para Molecular Craft, cuyo backend orquesta nodos (consulta, búsqueda en PubMed, agregación de respuesta) sobre un grafo de estados con memoria conversacional persistente, en consonancia con herramientas como LangGraph. Esta orquestación explícita facilita, además, la incorporación controlada de herramientas externas, aspecto que se aborda en el siguiente eje.

2.3. Model Context Protocol (MCP) e integración con herramientas externas

La capacidad de un agente para consultar proteínas o buscar literatura depende de su integración fiable con fuentes y herramientas externas. El Model Context Protocol (MCP) es un protocolo abierto que habilita la integración fluida entre aplicaciones basadas en LLM y fuentes de datos y herramientas externas (Model Context Protocol, 2025). Al estandarizar la forma en que un modelo descubre e invoca herramientas, MCP reduce el acoplamiento entre el agente y cada servicio particular, facilitando la incorporación de nuevas capacidades a la plataforma.

Desde su aparición, MCP se ha consolidado como un estándar práctico para conectar LLM con herramientas, archivos y APIs a través de aplicaciones anfitrionas heterogéneas (Model Context Protocol Team, 2025). Esta estandarización se enmarca en el campo más amplio del uso de herramientas (*tool use*) y la invocación de funciones (*function calling*). En lugar de acoplar el agente a cada servicio, MCP expone las herramientas de forma uniforme. Ello permite, en este trabajo, integrar la búsqueda en PubMed como una herramienta más del grafo del asistente.

2.4. Fundamentación en fuentes científicas verificables

Un riesgo central de los modelos de lenguaje es la generación de alucinaciones, es decir, afirmaciones plausibles pero incorrectas o no respaldadas por evidencia. Una línea de trabajo consolidada para mitigar este riesgo es la generación aumentada por recuperación (RAG), que incorpora en tiempo de inferencia información proveniente de fuentes externas. Las revisiones del área sistematizan estas técnicas —recuperación, aumento y generación— y sus límites (Fan et al., 2024; Zhang & Zhang, 2025).

El asistente desarrollado para Molecular Craft comparte el objetivo de RAG —fundamentar las respuestas en fuentes verificables— pero no implementa una arquitectura RAG basada en *embeddings* ni un índice vectorial. En su lugar, recupera literatura científica mediante la invocación de herramientas (*tool use*) a través de MCP sobre PubMed: el agente consulta la fuente, obtiene publicaciones reales y las vincula a la respuesta con su enlace de origen.



Esta distinción es deliberada y se enmarca en el Eje 3: el mecanismo es la integración con herramientas externas, no la recuperación densa de pasajes.

El valor de anclar el asistente en una fuente bibliográfica verificable como PubMed es doble. Por un lado, reduce el espacio de respuestas no fundamentadas, ya que cada afirmación relevante puede remitirse a una publicación concreta. Por otro, aumenta la trazabilidad del resultado y la confianza del usuario científico, al permitir verificar directamente la fuente. Así, el rigor no proviene de una métrica de reducción de alucinaciones medida en otro dominio, sino del diseño mismo: la respuesta se construye sobre referencias reales y consultables.

2.5. Síntesis

Los cuatro ejes se integran en torno al asistente agéntico como contribución central. El primero fundamenta, desde la economía de plataformas, por qué captar y retener una masa crítica de usuarios es estratégico para Molecular Craft. Aporta, además, una distinción que ordena el resto del trabajo: adoptar una plataforma y permanecer en ella responden a factores distintos, y la permanencia depende del valor percibido en el uso. Ahí se posiciona la asistencia inteligente como palanca, sin perder de vista el compromiso entre crecimiento y calidad de servicio.

Los tres ejes restantes fundamentan la robustez técnica del asistente. El Eje 2 aporta una arquitectura modular orquestada por grafos de estados. El Eje 3, su integración con herramientas externas mediante MCP. El Eje 4, la fundamentación de las respuestas en literatura científica verificable a través de PubMed, como vía para reducir alucinaciones. En conjunto, el marco teórico vincula la captación de usuarios con la solidez técnica del asistente, y sienta las bases del desarrollo descrito en los capítulos siguientes.

Nota sobre la jerarquía de la evidencia. Las fuentes empleadas en este capítulo cumplen funciones distintas y no tienen el mismo peso probatorio. Conviene explicitarlo, dada la rápida evolución del campo de los sistemas agénticos y de la integración de herramientas para modelos de lenguaje.

Las publicaciones arbitradas sustentan las afirmaciones conceptuales y analíticas centrales del marco. En esa categoría entran las revisiones sobre agentes basados en modelos de lenguaje, aprendizaje de herramientas, sistemas multiagente y alucinaciones. Entren también los estudios sobre efectos de red, competencia entre plataformas y, en particular, sobre adopción y continuidad de uso de plataformas SaaS, que fundamentan la dimensión de negocio del trabajo.

La documentación técnica primaria, los preprints recientes y las fuentes institucionales o de mercado cumplen, en cambio, la función de antecedentes complementarios. En esa categoría entran la especificación del Model Context Protocol, la documentación de los marcos de orquestación y las cifras de las plataformas de referencia. Describen estándares, herramientas, magnitudes del sector y decisiones de implementación. No se utilizan por sí solas para sustentar afirmaciones teóricas concluyentes.



3. Metodología

El capítulo anterior estableció los fundamentos conceptuales del trabajo: los efectos de red y la masa crítica, las arquitecturas agénticas, el Model Context Protocol y la fundamentación en fuentes científicas verificables. Corresponde ahora precisar cómo se traducen en un artefacto concreto. En este capítulo se describe la metodología empleada para el diseño, la implementación y la validación del asistente agéntico de bioinformática, así como para la migración a un entorno web de una sección de Molecular Craft. El aporte central de este trabajo es la construcción de un artefacto de software funcional, concebido como un producto mínimo viable (MVP) de carácter piloto. La metodología se orienta, por tanto, al desarrollo. Se estructura en fases que se corresponden de forma directa con los objetivos específicos planteados en el Capítulo 1.

3.1. Enfoque general

Esta investigación debe entenderse como un estudio aplicado de alcance exploratorio orientado al diseño y validación funcional de un artefacto de software. En consecuencia, sus resultados permiten demostrar factibilidad técnica y valor potencial de uso, pero no constituyen todavía evidencia concluyente de impacto sostenido en adopción, retención o desempeño comercial del modelo SaaS. Para abordar este objetivo, se articulan tres dimensiones de trabajo:

- Una dimensión de **ingeniería**, correspondiente al diseño de la arquitectura agéntica y a su implementación sobre un grafo de estados con memoria conversacional persistente.
- Una dimensión de **integración**, asociada a la incorporación de fuentes científicas externas —la API del Protein Data Bank y un servidor MCP de PubMed— como herramientas invocables por el agente (Model Context Protocol, 2025).
- Una dimensión de **demostración**, orientada a ilustrar —mediante una comparación entre el modelo de escritorio actual y la nueva versión web asistida— de qué manera la asistencia agéntica puede mejorar la experiencia de uso y, con ello, favorecer la adopción de la plataforma.

El resultado esperado es un asistente operativo, integrado en la sección de búsqueda del Protein Data Bank de Molecular Craft web. Debe ser capaz de responder consultas sobre proteínas, recuperar literatura científica verificable y orientar proactivamente al usuario en la continuación de su búsqueda.

3.2. Alcance y delimitación

Dado su carácter de piloto orientado a validar un concepto, el alcance se acota deliberadamente a una única sección de la aplicación: la **búsqueda en el Protein Data Bank (PDB)**. Esta sección se migra a un entorno web y se enriquece con el asistente, mientras que el resto



de los módulos de la plataforma se mantienen como referencia visual sin reimplementarse en su totalidad. Esta decisión responde a la naturaleza del aporte: el objetivo no es reconstruir la herramienta de escritorio completa, sino validar, sobre un caso concreto y representativo, un concepto replicable al resto de las secciones.

El asistente se concibe, además, como un agente *especializado* por sección. Su conocimiento y comportamiento se orientan al dominio de la consulta de proteínas y la búsqueda bibliográfica asociada, en lugar de cubrir de forma genérica todas las funcionalidades de la plataforma. Esta especialización es coherente con la arquitectura modular adoptada (Webb et al., 2025) y facilita la posterior réplica del enfoque en otros módulos.

3.3. Herramientas y tecnologías

La construcción del asistente se apoya en el conjunto de tecnologías descrito a continuación, seleccionadas por su madurez, su soporte para arquitecturas agénticas y su capacidad de integración con herramientas externas:

- **Lenguaje y servicio web:** Python 3.12 con FastAPI y Uvicorn para exponer la API del backend, empleando transmisión por eventos del servidor (*Server-Sent Events*, SSE) para entregar las respuestas del agente de forma incremental y preservar la sensación de respuesta en tiempo real.
- **Orquestación agéntica:** LangChain (v1) y LangGraph para construir el grafo de estados que coordina los nodos del asistente (Liu et al., 2025).
- **Modelo de lenguaje:** Google Gemini en su variante *flash* (Gemini 2.5 Flash), elegido por su baja latencia y buen desempeño, factor crítico para no degradar la experiencia de usuario; el backend admite alternar el proveedor del modelo (Gemini u OpenAI) mediante configuración.
- **Memoria conversacional:** PostgreSQL como almacén de persistencia, utilizado por el *checkpoint* de LangGraph para conservar el estado de cada conversación entre turnos, asociándolo a un identificador de hilo (*thread*).
- **Integración de fuentes externas:** la API del Protein Data Bank para la consulta de proteínas y un servidor MCP de PubMed para la recuperación de literatura científica; esta última se incorpora mediante el Model Context Protocol (Model Context Protocol, 2025).
- **Empaquetado y despliegue:** Docker y Docker Compose para contenerizar el backend y la base de datos, facilitando una ejecución reproducible del entorno.

Entre las tecnologías anteriores, la elección del marco de orquestación agéntica merece una justificación específica, dada la diversidad de alternativas disponibles. El Cuadro 1 contrasta LangGraph (LangChain, 2025) con otros dos marcos ampliamente adoptados —AutoGen



(Microsoft, 2025) y CrewAI (CrewAI, 2025)— según los criterios más relevantes para este trabajo.

Cuadro 1: Comparación de marcos de orquestación agéntica según los criterios relevantes para el trabajo.

Criterio	LangGraph	AutoGen	CrewAI
Paradigma	Grafo de estados (nodos y aristas)	Conversación entre agentes	Roles y tareas (<i>crews</i>)
Control del flujo	Explícito y condicional	Emergente, conversacional	De alto nivel; <i>Flows</i> para flujos deterministas
Persistencia del estado	Estado tipado y <i>checkpoint</i> nativo (PostgreSQL)	En la conversación	Memoria integrada, menos granular
Trazabilidad	Alta (LangSmith)	Media (Studio)	Limitada
Aprendizaje	Alto	Medio	Bajo
Madurez / estado	v1.0; runtime por defecto de LangChain	En modo mantenimiento (Microsoft migró a su Agent Framework)	Joven, en rápido crecimiento
Adecuación al caso	Alta	Baja	Media

La comparación evidencia por qué se optó por LangGraph. El asistente se concibe como un grafo de estados explícito, con tres rasgos: un ruteo condicional determinista —consulta nueva o de seguimiento—, una ejecución en paralelo con posterior agregación y una memoria conversacional persistente. Este patrón coincide con el paradigma nativo de LangGraph: nodos, aristas condicionales y estado tipado. Coincide también con su soporte de persistencia mediante *checkpoint* sobre PostgreSQL.

Las alternativas se ajustan peor al caso. AutoGen organiza la coordinación como una conversación entre agentes, modelo menos adecuado para un flujo estructurado y determinista. Su desarrollo, además, se ha desacelerado tras el giro de Microsoft hacia su *Agent Framework*. CrewAI ofrece una abstracción de más alto nivel basada en roles, cómoda para equipos de agentes. Entrega, sin embargo, menor control explícito del flujo y de la persistencia granular que este caso requiere.

La mayor curva de aprendizaje de LangGraph se consideró un costo asumible. A cambio, se obtiene control sobre el flujo, trazabilidad e idoneidad para la arquitectura propuesta.

Además del marco de orquestación, la elección del modelo de lenguaje responde a un criterio explícito de latencia. Dado que el asistente entrega sus respuestas de forma incremental por SSE, la velocidad de generación condiciona directamente la sensación de respuesta en tiempo real. El Cuadro 2 contrasta Gemini 2.5 Flash con GPT-4o mini —el modelo del proveedor alternativo soportado por el backend— según mediciones públicas de la plataforma



independiente de benchmarking Artificial Analysis (Artificial Analysis, 2026) y las tarifas oficiales de cada API.

Cuadro 2: Comparación de modelos de lenguaje: velocidad de salida (Artificial Analysis) y tarifas de API.

Métrica	Gemini 2.5 Flash	GPT-4o mini
Velocidad de salida (tokens/s)	~204	~85
Precio de entrada (USD / 1M tokens)	0,30	0,15
Precio de salida (USD / 1M tokens)	2,50	0,60
Ventana de contexto	1M	128K
Corte de conocimiento	Ene. 2025	Oct. 2023

La comparación evidencia un compromiso deliberado. Gemini 2.5 Flash genera texto a una velocidad cercana a 2,4 veces la de GPT-4o mini, lo que favorece la experiencia interactiva por *streaming* que persigue el asistente, a cambio de un mayor costo por token de salida. En esta etapa de MVP, el criterio prioritario es preservar la sensación de respuesta en tiempo real, no minimizar el costo. Se optó, por ello, por Gemini 2.5 Flash. Se conserva, eso sí, la posibilidad de alternar el proveedor por configuración cuando el costo o la disponibilidad lo justifiquen. Las cifras corresponden a la fecha de consulta y pueden variar con las nuevas versiones de los modelos.

3.4. Fases de trabajo

El desarrollo se organizó en cinco fases secuenciales, cada una asociada a un objetivo específico (OE). El Cuadro 3 resume esta correspondencia.



Cuadro 3: Correspondencia entre las fases de trabajo y los objetivos específicos.

Fase	Descripción	Objetivo
Fase 1	Migración de la sección de búsqueda en el Protein Data Bank a un entorno web.	OE1
Fase 2	Diseño e implementación de la arquitectura agéntica: grafo de estados con memoria conversacional persistente.	OE2
Fase 3	Integración de fuentes científicas externas: API del PDB y servidor MCP de PubMed.	OE3
Fase 4	Implementación del comportamiento proactivo: recuperación de literatura, vinculación de publicaciones y propuesta de nuevas búsquedas.	OE4
Fase 5	Demostración del asistente como mecanismo de captación y <i>engagement</i> de usuarios.	OE5

3.4.1. Fase 1: Migración de la búsqueda PDB a un entorno web

Se reimplementó en un entorno web la sección de búsqueda del Protein Data Bank, originalmente disponible solo en la aplicación de escritorio. La interfaz expone tres modos de búsqueda: por texto completo, por identidad de secuencia y por atributo. Incorpora, además, un control del número de resultados por consulta y un listado de proteínas candidatas, ordenable por nivel de similitud, desde el cual el usuario selecciona la proteína de interés. El diseño priorizó la usabilidad, incorporando descriptores y leyendas explicativas en cada campo de búsqueda para que el propósito de cada control resulte evidente al usuario.

3.4.2. Fase 2: Diseño e implementación de la arquitectura agéntica

El asistente se construyó sobre un grafo de estados explícito, paradigma que permite coordinar de forma estructurada un conjunto de nodos especializados (Liu et al., 2025). Cada nodo asume una responsabilidad acotada —consulta de proteínas, búsqueda bibliográfica en PubMed y agregación de la respuesta final—, y todos comparten un estado común que se propaga a lo largo de la ejecución. Para dotar al asistente de memoria conversacional, el estado del grafo se persiste mediante el *checkpointing* sobre PostgreSQL, lo que permite retomar el contexto de una conversación entre turnos y conservar las publicaciones ya recuperadas. El diseño se abordó de manera iterativa, refinando la división de responsabilidades y los contratos de datos entre nodos a medida que se incorporaban nuevas capacidades.



3.4.3. Fase 3: Integración de fuentes científicas externas

Esta fase conectó el asistente con las dos fuentes verificables de su dominio, que consulta en tiempo de ejecución mediante la invocación de herramientas (*tool use*): el Protein Data Bank, para los metadatos de proteínas, y un servidor MCP de PubMed, para la literatura científica (Model Context Protocol, 2025). Con ello, las referencias que entrega el asistente no dependen del conocimiento interno del modelo, sino que remiten a artículos reales y consultables. La arquitectura concreta de esta integración se detalla en el Capítulo 4.

3.4.4. Fase 4: Implementación del comportamiento proactivo

En esta fase se dotó al asistente de un comportamiento proactivo, orientado a anticipar la siguiente necesidad del usuario antes que a limitarse a responder cada mensaje. La pieza que lo habilita es la memoria de publicaciones que el estado del grafo acumula por conversación: permite reutilizar resultados en las consultas de seguimiento, evitar búsquedas redundantes y respaldar la respuesta cuando una nueva búsqueda no aporta información. A partir de esa memoria, el asistente propone de forma activa rutas de continuación —un resumen de los artículos, la ampliación de la lista o el detalle de un resultado—, guiando la exploración sin perder el contexto de la conversación.

3.4.5. Fase 5: Demostración como mecanismo de captación

Finalmente, una demostración comparativa contrasta la versión de escritorio con la nueva versión web asistida. Con ella se ilustra el potencial del asistente como mecanismo de captación y *engagement*, evidenciando las diferencias en la experiencia de uso y el valor agregado de la asistencia inteligente. Esta dimensión conecta el aporte técnico con el objetivo general del trabajo: aportar valor desde el primer uso para acelerar la incorporación de usuarios. Se alinea así con la noción de masa crítica y efectos de red en plataformas digitales (Cervini & Joglekar, 2025).

3.5. Criterios de validación

La validación del trabajo se aborda en tres planos complementarios, coherentes con su carácter de MVP:

- **Validación funcional:** verificación de que los flujos del asistente entregan la salida esperada y de que el contrato de comunicación (SSE) se mantiene estable. Se apoya en pruebas automatizadas (*pytest*) que cubren la construcción del grafo, el comportamiento de la memoria y el ruteo de las consultas de seguimiento.
- **Validación de comportamiento:** comprobación de que el asistente reutiliza la memoria de publicaciones en consultas de seguimiento, aplica el mecanismo de respaldo cuando no hay resultados nuevos y propone rutas de continuación pertinentes.

- **Demostración comparativa:** contraste cualitativo entre el flujo de escritorio y el flujo web asistido, registrando las diferencias relevantes en usabilidad y propuesta de valor.

Estos criterios privilegian la evidencia de funcionamiento del artefacto por sobre la medición numérica de desempeño. Ello es consistente con el objetivo de demostrar la viabilidad del asistente y de explorar su rol potencial como palanca de adopción. La definición de métricas formales de *engagement* se aborda, en cambio, en la Sección 3.6.

Corresponde explicitar que la ausencia de validación con usuarios reales es una decisión de diseño y no una omisión. El objetivo de esta etapa es construir y evidenciar el funcionamiento de un producto mínimo viable, no medir su efecto sobre la adopción.

Una evaluación con investigadores exigiría cuatro elementos: un artefacto estabilizado, un protocolo de tareas, un instrumento de usabilidad validado y una ventana de observación longitudinal. Su costo excede el alcance de este trabajo. Solo resultan informativos, además, una vez que el asistente opera en contexto real de uso.

Se asume, por tanto, un compromiso deliberado (*trade-off*). Se privilegia la evidencia de factibilidad técnica y de valor funcional del artefacto, verificable de forma reproducible. La evidencia empírica de captación y retención queda planteada como hipótesis a contrastar y como primera línea de trabajo futuro (Capítulo 5). Este compromiso acota lo que el trabajo permite concluir. Delimita también, con precisión, lo que habilita: el propio asistente provee el instrumento de medición necesario para cerrar esa brecha.

3.6. Diseño de la validación futura

Los criterios anteriores delimitan lo que este trabajo valida; corresponde ahora precisar cómo debiera validarse aquello que queda fuera de su alcance, de modo que la brecha empírica no se enuncie solo como pendiente, sino como un diseño ejecutable. Se propone, en consecuencia, una siguiente etapa de validación articulada en dos estudios complementarios.

El primero es un **estudio de usabilidad y comparación con el flujo de escritorio**, de carácter intra-sujeto. Participa en él un grupo reducido de usuarios del perfil objetivo: investigadores o estudiantes de posgrado en química, bioquímica o bioinformática. Cada participante ejecuta un conjunto equivalente de tareas en ambos flujos, con el orden contrabalanceado para controlar el efecto de aprendizaje. De cada sesión se registran el tiempo de tarea, la tasa de finalización y, al cierre, la Escala de Usabilidad del Sistema (SUS), el cuestionario estandarizado más utilizado para medir la usabilidad percibida (Lewis, 2018). El segundo es un **estudio longitudinal de uso** sobre una cohorte de usuarios reales, con una ventana de observación de varias semanas. Mide el *engagement*, la profundidad de uso y la retención entre sesiones: variables que solo resultan informativas cuando el asistente opera en contexto real.

Este diseño se apoya, además, en la literatura sobre evaluación de agentes basados en modelos de lenguaje. Dicha literatura advierte que un agente no puede evaluarse como un modelo aislado. Al operar en entornos interactivos —donde planifica, invoca herramientas y usa memoria—, su valoración exige combinar métricas automáticas con evaluación humana en contexto de uso (Mohammadi et al., 2025). De ahí que el diseño propuesto no se agote en las pruebas automatizadas ya ejecutadas, sino que incorpore usuarios reales.



El Cuadro 4 sistematiza las métricas propuestas y el instrumento o la fuente de datos que permitiría obtener cada una.

Cuadro 4: Métricas, instrumentos y fuentes de datos propuestos para la validación futura del asistente.

Dimensión	Métrica	Instrumento o fuente de datos
Usabilidad percibida	Puntaje de usabilidad del sistema (SUS, escala 0–100) (Lewis, 2018)	Cuestionario estandarizado de diez ítems aplicado al término de la sesión
Eficiencia	Tiempo de tarea: localizar una proteína y su literatura pertinente	Registro temporal de la sesión, sobre un protocolo de tareas idénticas en ambos flujos
Eficacia	Tasa de finalización de la tarea sin asistencia externa	Observación estructurada y registro de eventos del <i>backend</i>
<i>Engagement</i>	Tasa de continuación: sugerencias proactivas aceptadas sobre respuestas entregadas	Eventos [SUGGESTIONS] del flujo SSE y número de turnos por hilo
Profundidad de uso	Publicaciones consultadas y turnos de conversación por sesión	Memoria de publicaciones y estado persistido en PostgreSQL
Retención	Proporción de usuarios que retornan en la segunda y la cuarta semana	Identidad de usuario asociada al hilo de conversación (requiere autenticación)
Comparación con el escritorio	Diferencia de tiempo de tarea y de tasa de finalización entre ambos flujos	Diseño intra-sujeto con tareas contrabalanceadas

Este diseño tiene una condición previa que conviene declarar: exige incorporar la instrumentación de la que el prototipo carece —identidad de usuario, telemetría de eventos y observabilidad—, señalada entre sus limitaciones técnicas en el Capítulo 4. Dicho de otro modo: el propio asistente provee la materia prima de la medición, pues cada interacción queda registrada como dato estructurado. Convertirla en métricas exige, sin embargo, una capa analítica. Esa tarea se recoge como primera línea de trabajo futuro (Capítulo 5).



4. Desarrollo

Este capítulo presenta el desarrollo del asistente agéntico de bioinformática y de la versión web de la búsqueda en el Protein Data Bank, siguiendo el orden de los objetivos específicos y las fases definidas en el capítulo anterior. Se describe primero la arquitectura general de la solución y la migración de la sección a un entorno web. Se detalla luego el backend agéntico: su organización, el grafo de estados, la memoria persistente y la API. Siguen la integración con fuentes científicas externas, el comportamiento proactivo del asistente y las pruebas realizadas. El capítulo cierra con una demostración comparativa entre ambas versiones y con la discusión de los resultados.

Para efectos de interpretación, conviene fijar de antemano una clave de lectura: los resultados expuestos en este capítulo se distinguen en tres niveles. El primero corresponde a las **funcionalidades efectivamente implementadas**, verificables en el artefacto y respaldadas por las pruebas automatizadas. El segundo, a los **comportamientos observados** durante la demostración del prototipo —por ejemplo, la reutilización de la memoria de publicaciones o la trazabilidad de las referencias entregadas—. El tercero, a los **impactos potenciales esperados** en un escenario de operación real, que este trabajo no mide y que se enuncian, por tanto, como proyecciones. Esta diferenciación evita extrapolar más allá de la evidencia disponible y ordena la lectura de los apartados finales del capítulo.

4.1. Arquitectura general de la solución

La solución se estructura en tres componentes que se comunican entre sí. El **frontend web** presenta la sección de búsqueda en el Protein Data Bank junto a la interfaz de conversación del asistente. El **backend agéntico**, implementado en Python con FastAPI, orquesta el asistente sobre un grafo de estados y entrega sus respuestas de forma incremental mediante transmisión por eventos del servidor (SSE). Finalmente, las **fuentes científicas externas** —la API del Protein Data Bank y un servidor MCP de PubMed— proveen, respectivamente, la información de proteínas y la literatura asociada. La memoria conversacional se conserva en una base de datos PostgreSQL, de modo que cada conversación mantiene su contexto entre turnos. La Figura 7 resume esta organización.

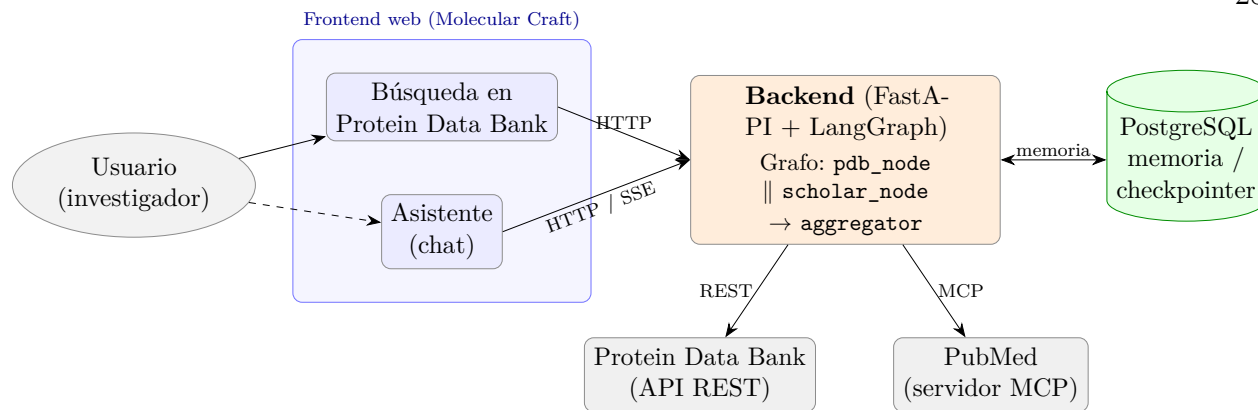


Figura 7: Arquitectura general de la solución.

4.2. Migración de la búsqueda en el Protein Data Bank

La sección de búsqueda en el Protein Data Bank, originalmente disponible solo en la aplicación de escritorio, se reimplementó como una aplicación web. Esta migración constituye la base sobre la cual se integra el asistente y materializa el primer objetivo específico: llevar a un entorno accesible vía navegador una funcionalidad representativa de la plataforma.

4.2.1. Modos de búsqueda y parámetros

La interfaz permite consultar el Protein Data Bank mediante tres modos de búsqueda: por **texto completo**, por **identidad de secuencia** y por **atributo**. El presente trabajo se centra en la búsqueda por texto completo, que recibe el nombre o descriptor de una proteína y recupera las entradas relacionadas. La consulta admite, además, un parámetro que controla la **cantidad de resultados** por búsqueda. Atendiendo a la usabilidad, cada control incorpora descriptores y leyendas explicativas que aclaran su propósito al usuario, de modo que resulte evidente dónde ingresar la consulta y qué efecto tiene cada opción. La Figura 8 muestra la interfaz de búsqueda.



Figura 8: Interfaz web de búsqueda en el Protein Data Bank: parámetros de consulta y listado de resultados ordenado por *score*.

4.2.2. Presentación de resultados y acciones

Tras ejecutar la búsqueda, la aplicación presenta —en la misma interfaz (Figura 8)— un listado de proteínas candidatas ordenado por nivel de similitud (*score*) respecto de la consulta. Desde este listado, el usuario selecciona la proteína de interés, lo que dispara la generación del resumen y el inicio de la conversación con el asistente en el panel contiguo. El listado dispone, asimismo, de acciones complementarias: filtrar los resultados, descargarlos y eliminarlos de la vista. El resultado de seleccionar una proteína e interactuar con el asistente se muestra más adelante, en la Figura 11.

4.3. Backend agéntico

El backend concentra la lógica del asistente: expone la API consumida por el frontend, orquesta el grafo de estados que coordina los nodos del agente y gestiona la memoria conversacional persistente.

4.3.1. Organización del proyecto

El backend se organiza en módulos con responsabilidades acotadas. El punto de entrada (`main.py`) define la aplicación FastAPI y sus *endpoints*. Un módulo de memoria



(`memory.py`) gestiona la conexión con el *checkpoint* sobre PostgreSQL. Un módulo de esquemas (`models/schemas.py`) define los contratos de datos mediante Pydantic. Por último, un paquete dedicado (`graph/`) contiene la construcción del grafo y la implementación de sus nodos. El proyecto se conteneriza con Docker y Docker Compose, lo que permite levantar de forma reproducible tanto el servicio del backend como la base de datos.

4.3.2. Grafo de estados y nodos

El asistente se implementa como un grafo de estados explícito construido con LangGraph, paradigma que extiende los flujos de trabajo basados en grafos a los agentes basados en modelos de lenguaje y habilita una ejecución estructurada (Liu et al., 2025). A partir de un despacho condicional en el nodo inicial, el grafo coordina tres nodos especializados:

- Un **nodo PDB** (`pdb_node`), que resuelve los metadatos de la proteína —nombre, método experimental, resolución e identificador— consultando la API REST del Protein Data Bank (RCSB); admite además resolver varios identificadores en paralelo (búsqueda por lotes).
- Un nodo de **búsqueda bibliográfica** (`scholar_node`), que recupera publicaciones científicas reales asociadas a la proteína desde PubMed a través de un servidor MCP.
- Un nodo **agregador** (`aggregator_node`), que sintetiza la información de los dos nodos anteriores mediante el modelo de lenguaje, genera el resumen y la respuesta final, y aplica reglas de formato según el tipo de consulta.

El despacho no es secuencial: desde el inicio, el grafo realiza un *fan-out* en paralelo hacia el nodo PDB y el nodo de búsqueda bibliográfica, y el agregador espera a que ambos finalicen antes de componer la respuesta. El ruteo es condicional. En una consulta de seguimiento que ya cuenta con memoria suficiente, el grafo omite el nodo de búsqueda bibliográfica, y evita así consultas redundantes a PubMed. El nodo PDB, en cambio, se ejecuta en todos los casos. La clasificación del turno —consulta nueva o de seguimiento— es determinista, basada en señales lingüísticas y en la memoria disponible. Todos los nodos comparten un estado común que se propaga a lo largo de la ejecución y que acumula, entre otros elementos, las publicaciones recuperadas durante la conversación (este mecanismo se detalla en la Sección 4.5). La Figura 9 presenta el grafo del asistente.

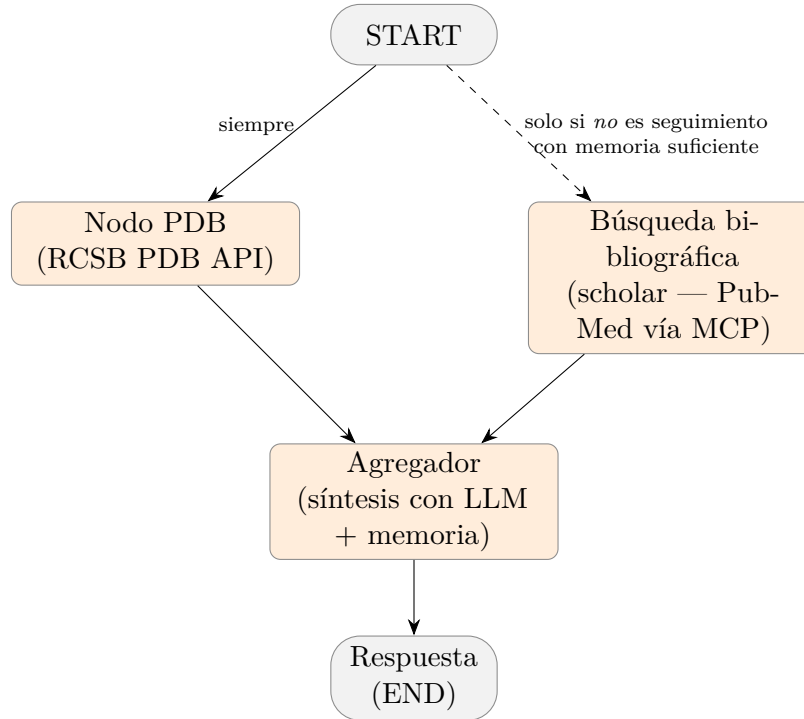


Figura 9: Grafo de estados del asistente y sus nodos.

4.3.3. Memoria conversacional persistente

Para dotar al asistente de memoria, el estado del grafo se persiste mediante el *checkpoint* de LangGraph sobre PostgreSQL. Cada conversación se identifica con un *thread* (`thread_id`), lo que permite retomar su contexto entre turnos sin perder la información previa. De esta manera, el asistente conserva una memoria de corto plazo de la conversación, del orden de las últimas tandas de intercambio. Mantiene, en particular, una memoria de las publicaciones ya recuperadas, clave para el comportamiento proactivo descrito más adelante.

4.3.4. API y contrato de *streaming*

El backend expone un conjunto de *endpoints* REST, resumidos en el Cuadro 5. Las respuestas del asistente se entregan mediante transmisión por eventos del servidor (SSE), de modo que el texto se muestra de forma incremental a medida que se genera, preservando la sensación de respuesta en tiempo real.



Cuadro 5: Principales *endpoints* expuestos por el backend.

Endpoint	Descripción
GET /health	Verificación de estado del servicio.
POST /agent/stream	Respuesta del agente por SSE (entrada: consulta).
GET /agent/modal	Contenido y resumen de una proteína por SSE.
POST /agent/scholar	Búsqueda de publicaciones en PubMed por proteína.
POST /conversations/start	Crea un <code>thread_id</code> para una nueva conversación.
POST /conversations/{thread_id} /send	Envía un mensaje y responde por SSE con memoria.
GET /conversations/{thread_id} /messages	Historial de mensajes de la conversación.
POST /agent/stream/legacy	<i>Endpoint</i> heredado, compatible con el formato anterior.

El contrato de los flujos SSE es estable y emplea los siguientes tipos de evento:

- `data`: `<fragmento>` para cada porción de texto incremental.
- `data`: `[DONE]` al finalizar la respuesta.
- `data`: `[ERROR] <mensaje>` ante una condición de error.
- En el flujo conversacional, las sugerencias proactivas se entregan en un evento `data`: `[SUGGESTIONS] <json>`, separado del texto de la respuesta.

Adicionalmente, el backend valida las variables de entorno requeridas al iniciar —y se detiene si falta una clave de API necesaria— y habilita CORS para los orígenes del frontend, tanto locales como configurables.

4.4. Integración con fuentes científicas externas

Un principio de diseño del asistente es que la rigurosidad de sus respuestas no proviene del conocimiento interno del modelo, sino de fuentes verificables consultadas en tiempo de ejecución mediante la invocación de herramientas (*tool use*). La consulta de proteínas se realiza contra la API del Protein Data Bank, mientras que la búsqueda bibliográfica se delega



a un servidor MCP de PubMed a través del Model Context Protocol (Model Context Protocol, 2025). A diferencia de una arquitectura de generación aumentada por recuperación basada en *embeddings*, el asistente no construye un índice vectorial de pasajes, sino que invoca PubMed como una herramienta más del grafo (Zhang & Zhang, 2025).

El beneficio de este enfoque es doble. Por un lado, cada publicación citada corresponde a un artículo real, vinculado a la respuesta mediante su enlace de origen, de modo que el usuario puede verificar la fuente. Por otro, se reduce por diseño el espacio de afirmaciones no fundamentadas. Es una ventaja relevante frente a los asistentes que devuelven referencias plausibles, pero no siempre comprobables.

4.5. Comportamiento proactivo del asistente

Sobre la base anterior se implementó el comportamiento proactivo del asistente, entendido como su capacidad de anticiparse a las necesidades del usuario en lugar de limitarse a reaccionar a sus mensajes. El mecanismo se apoya en la memoria de publicaciones (*paper memory*) que el estado del grafo acumula por conversación, y se rige por la siguiente lógica:

- En una **consulta nueva**, el agregador prioriza los resultados frescos del nodo de búsqueda bibliográfica y compone la respuesta aplicando reglas de formato propias de este caso (un resumen, una tabla de publicaciones y sus enlaces).
- En una **consulta de seguimiento** con memoria suficiente, el ruteo puede omitir el nodo de búsqueda para evitar consultas redundantes a PubMed, reutilizando las publicaciones ya recuperadas.
- Si en un turno no se obtienen resultados nuevos, el agregador recurre a un mecanismo de respaldo (*fallback*) sobre la memoria de publicaciones, de modo que la respuesta no quede vacía.

Este comportamiento es parametrizable. Existe un límite máximo de publicaciones conservadas por conversación. Existe, asimismo, un umbral mínimo de publicaciones en memoria a partir del cual una consulta de seguimiento puede prescindir de una nueva búsqueda. La memoria fusiona, además, las publicaciones, evitando duplicados y conservando las más recientes.

Más allá de la gestión de la memoria, el asistente orienta activamente la exploración del usuario. Junto a cada respuesta anexa entre dos y cuatro sugerencias contextuales. Al menos una recomienda una nueva búsqueda —proteínas relacionadas, homólogos o inhibidores, por ejemplo—; las restantes proponen preguntas de seguimiento específicas. Todas guían la continuación del trabajo sin perder el contexto de la conversación. De este modo, el asistente se aproxima a la noción de agente anticipatorio, que planifica y sugiere acciones orientadas a metas en lugar de responder de forma puramente reactiva (Webb et al., 2025). Refuerza así la experiencia de uso que este trabajo busca convertir en palanca de adopción (Cervini & Joglekar, 2025).



4.6. Pruebas

El funcionamiento del *backend* se respalda con un conjunto de 37 pruebas automatizadas ejecutadas con *pytest*, que se completan satisfactoriamente y verifican de forma reproducible los criterios funcionales y de comportamiento definidos en la metodología. Las pruebas se organizan en dos niveles. Las pruebas **unitarias** aíslan cada pieza de la lógica del asistente: la construcción del grafo, el cliente de integración con el Protein Data Bank, la resolución de proteínas y la gestión de la memoria de publicaciones. Las pruebas de **integración de flujo**, por su parte, ejercitan la coordinación entre nodos, desde la consulta hasta la respuesta agregada. El Cuadro 6 resume los grupos de pruebas y el aspecto que valida cada uno.

Cuadro 6: Grupos de pruebas automatizadas del *backend* del asistente.

Grupo de pruebas	Cantidad	Aspecto validado
Grafo del asistente	12	Construcción del grafo, nodos y transiciones, y ejecución del flujo conversacional
Cliente PDB/RCSB	6	Resolución de identificadores y metadatos desde la API del Protein Data Bank
Resolución por lotes	8	Consulta simultánea de múltiples proteínas
Flujo PDB–bibliografía	3	Integración: consulta de proteína, búsqueda de publicaciones y agregación de la respuesta
Memoria y ruteo	8	Reutilización de publicaciones en memoria, ruteo de las consultas de seguimiento y mecanismo de respaldo
Total	37	

Como criterio de aceptación se estableció que la totalidad de las pruebas se completara satisfactoriamente, condición que se cumple en la ejecución del *suite*.

Para cuantificar el alcance de estas pruebas se midió la cobertura de código con *pytest-cov*. La lógica central del asistente comprende la construcción del grafo y su estado, los nodos especializados, el cliente de integración con el Protein Data Bank y los esquemas de datos. Sobre esos módulos, la cobertura alcanza un **76 %**, con varios de ellos por sobre el 95 % (Cuadro 7). La capa de servicio web (*endpoints* y transmisión por SSE) y el *checkpoint* de persistencia no forman parte de esta medición unitaria; su funcionamiento se evidencia en el uso del sistema. Al considerarlas, la cobertura global del *backend* se sitúa en un **46 %**.



Cuadro 7: Cobertura de código por componente (pytest-cov).

Componente	Sentencias	Cobertura
Construcción del grafo y estado	109	97 %
Nodos del asistente	279	62 %
Cliente de integración PDB (RCSB)	64	83 %
Modelos y esquemas de datos	53	100 %
Lógica central del asistente (subtotal)	505	76 %
Capa de API/SSE y persistencia	330	0 %
Total del <i>backend</i>	835	46 %

El Cuadro 8 sintetiza los criterios de aceptación definidos y, sobre todo, delimita explícitamente qué quedó validado por la evidencia recogida y qué no.

Cuadro 8: Criterios de aceptación y alcance efectivo de la validación realizada.

Dimensión	Criterio de aceptación	Estado
Construcción del grafo	El grafo se construye con sus nodos y transiciones, y el flujo conversacional se ejecuta de extremo a extremo.	Validado
Integración PDB	Los identificadores y metadatos se resuelven contra la API del Protein Data Bank, incluida la consulta por lotes.	Validado
Integración PubMed (MCP)	Cada publicación devuelta corresponde a un artículo real, vinculado a su enlace de origen.	Validado
Memoria y ruteo	Las consultas de seguimiento reutilizan la memoria de publicaciones y el mecanismo de respaldo evita respuestas vacías.	Validado
Suite de pruebas	La totalidad de las 37 pruebas automatizadas se completa satisfactoriamente.	Validado
Cobertura de la lógica central	Cobertura ≥ 70 % sobre los módulos que implementan la lógica del asistente (obtenida: 76 %).	Validado
Capa de API/SSE y persistencia	Sin pruebas unitarias (0 %); su funcionamiento se evidencia solo en el uso del sistema.	No validado
Usabilidad y experiencia de uso	Sin métricas de tiempo de tarea, satisfacción percibida ni tasa de finalización.	No validado
Captación y retención	Sin medición de continuidad de uso, conversión ni retención entre sesiones.	No validado
Desempeño bajo carga	Sin pruebas de concurrencia, latencia sostenida ni degradación del servicio.	No validado

Las pruebas implementadas permiten verificar el funcionamiento correcto de componentes



críticos del *backend* y del flujo agéntico; sin embargo, su alcance se circunscribe a la validación técnica del artefacto. En trabajos futuros será necesario complementar esta evidencia con validación orientada al usuario, incluyendo métricas de usabilidad, tiempos de tarea, satisfacción percibida y continuidad de uso.

4.7. Limitaciones técnicas del prototipo

En coherencia con su condición de producto mínimo viable, el artefacto presenta limitaciones técnicas que conviene explicitar. Se agrupan en tres planos: operación, funcionalidad y pruebas.

En el plano de la **operación**, el *backend* no incorpora autenticación, control de acceso ni separación por organización (*multi-tenancy*). Son elementos indispensables en un esquema de suscripción real. Tampoco implementa límites de tasa sobre las llamadas al modelo de lenguaje ni a las fuentes externas. Falta, asimismo, instrumentación de observabilidad: trazas, métricas y alertas. El despliegue se resuelve con Docker y Docker Compose en un entorno local reproducible. No se aborda, en cambio, una configuración productiva: escalamiento, gestión de secretos, respaldos ni disponibilidad.

En el plano **funcional**, la clasificación del turno como consulta nueva o de seguimiento es determinista y se apoya en señales lingüísticas. Resulta, por ello, frágil ante fraseos ambiguos que un clasificador basado en modelo de lenguaje manejaría con mayor flexibilidad. La fundamentación bibliográfica, por su parte, queda acotada a lo que PubMed indexa y a la calidad de la consulta formulada. Depende, además, de la disponibilidad de un servicio externo.

En el plano de las **pruebas**, la cobertura se concentra en la lógica central del asistente y deja fuera la capa de servicio web y la persistencia.

Ninguna de estas limitaciones compromete la validación del concepto que persigue el trabajo. Todas, sin embargo, delimitan la distancia entre el prototipo y una versión operable en producción.

4.8. Demostración: comparación escritorio *vs.* web asistida

Para evidenciar el valor de la solución se realiza una demostración comparativa entre la versión de escritorio de la sección y la nueva versión web asistida. La versión de escritorio se limita a recuperar y presentar entradas del Protein Data Bank. La versión web, en cambio, incorpora un asistente que resume la proteína seleccionada, recupera literatura real desde PubMed con sus enlaces de origen y guía proactivamente la continuación de la búsqueda. Las Figuras 10 y 11 contrastan ambos flujos.



37

0 Molecular Craft - Búsqueda PDB

Inicio Búsqueda PDB

Ingresar sesión a buscar

Selección una sesión de investigación

Búsqueda en Protein Data Bank

Tipo de búsqueda
Búsqueda de texto completo

Número máximo de resultados de búsqueda
10

Valor de búsqueda
hsp90

Buscar

Resultados	Descargar
1SF8	Descargar PDB
7K3R	Descargar PDB
7K3U	Descargar PDB
7K3W	Descargar PDB
3K6D	Descargar PDB
2G1T	Descargar PDB
7K3S	Descargar PDB
7K3V	Descargar PDB
3QSL	Descargar PDB
2ICP	Descargar PDB

Figura 10: Flujo de búsqueda en la versión de escritorio.

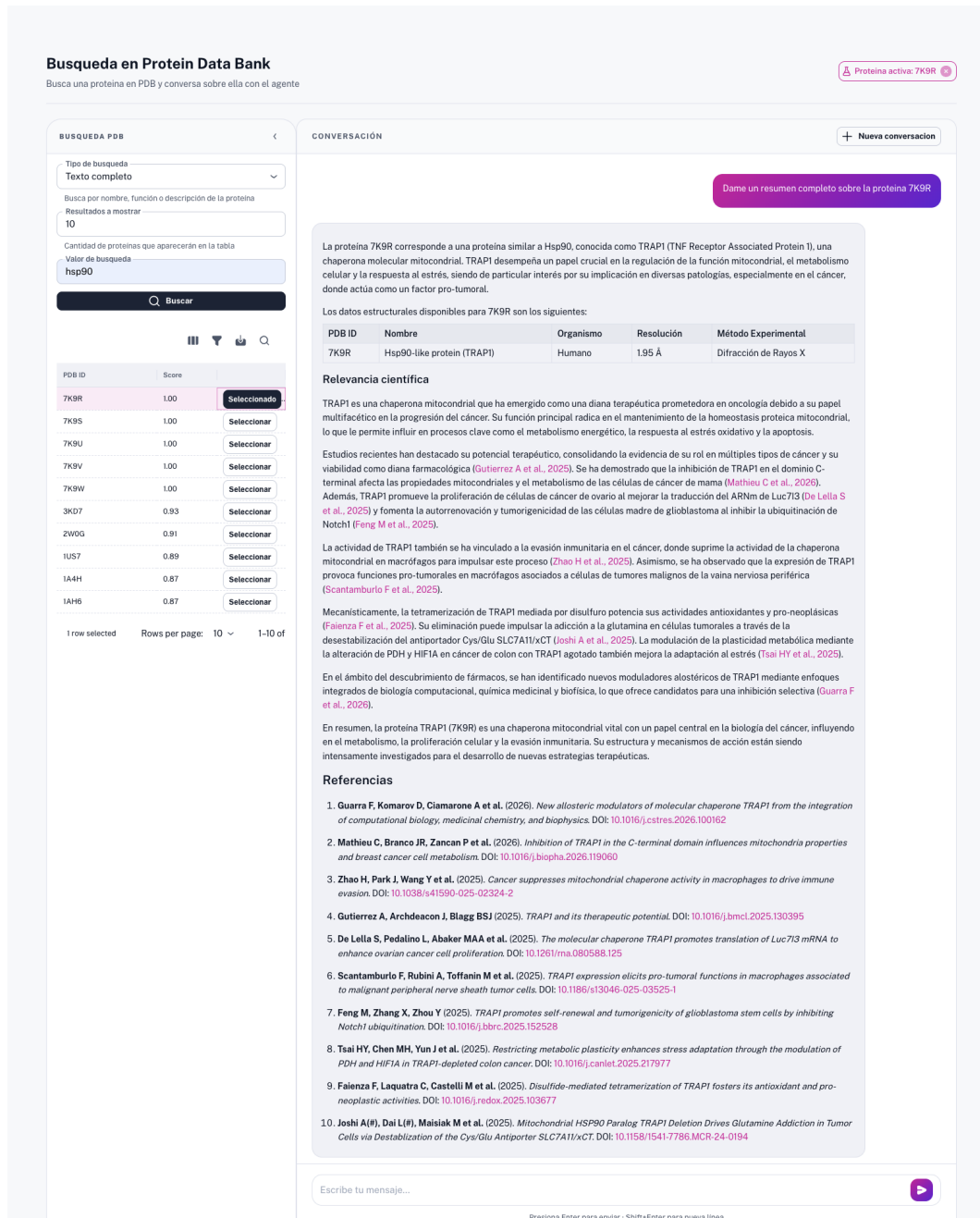


Figura 11: Flujo de búsqueda en la versión web asistida.

La comparación entre la experiencia de escritorio y la experiencia web asistida permite mostrar de manera ilustrativa cómo la asistencia inteligente reduce fricción, mejora la orientación del usuario y agrega trazabilidad sobre la búsqueda científica. No obstante, estos resultados deben leerse como evidencia demostrativa del potencial del enfoque, más que como validación definitiva de captación o retención, lo que requerirá medición en contexto real de uso.



4.9. De la solución técnica al resultado estratégico: cadena de valor

Los apartados anteriores describen el artefacto y su funcionamiento; conviene ahora hacer explícito el puente entre la solución técnica y el resultado estratégico que persigue el trabajo. Este puente puede sintetizarse como una cadena de valor de cuatro eslabones, planteada como la lógica que articula la propuesta y no como una relación ya demostrada empíricamente:

1. La asistencia inteligente reduce la fricción y aumenta la utilidad percibida desde la primera interacción (valor desde el primer uso).
2. Esa mejora en la experiencia inicial incrementa la probabilidad de adopción y de continuidad de uso (Cao et al., 2026).
3. Cada interacción mediada por el asistente genera datos estructurados sobre necesidades, intereses y patrones de consulta.
4. Dichos datos habilitan, en operación real, decisiones de mejora del producto, segmentación y diseño de mecanismos de captación y retención más precisos, que realimentan el primer eslabón.

Conviene precisar el estatuto de cada eslabón. El presente trabajo evidencia la base del primero: la funcionalidad está disponible desde la primera interacción, sin instalación ni configuración previas. Deja instalado, asimismo, el mecanismo que produce el tercero, esto es, la captura de datos estructurados.

Que esa funcionalidad se traduzca efectivamente en valor percibido es, en cambio, un efecto esperado. Otro tanto ocurre con los eslabones segundo y cuarto. El vínculo entre valor percibido y continuidad de uso no es, con todo, una conjetura de este trabajo: la literatura arbitrada sobre productos SaaS por suscripción lo documenta de forma consistente (Cao et al., 2026). Lo que aquí falta no es el fundamento del mecanismo, sino su medición en este caso concreto. Esa medición se aborda conforme al diseño propuesto en la Sección 3.6, en línea con la lógica de masa crítica y efectos de red que enmarca el trabajo (Cervini & Joglekar, 2025). Esta cadena ordena, además, la lectura de los beneficios por tipo de *stakeholder* que se presenta a continuación.

4.10. Beneficios por tipo de stakeholder

Los beneficios de la solución pueden sistematizarse según el actor al que impactan. Se distinguen dos clases. Los beneficios **directos** se evidencian de inmediato en el uso del prototipo. Los beneficios **esperados** o **potenciales** se proyectan para un escenario de operación real y derivan del uso sostenido y de la escala. La distinción es deliberada: evita presentar como resultados demostrados aquello que son consecuencias plausibles, pero aún no medidas. El Cuadro 9 resume esta correspondencia.



Cuadro 9: Beneficios directos (evidenciados) y esperados de la solución por tipo de *stakeholder*.

Stakeholder	Beneficios	Impacto esperado
Usuario científico (investigador)	Respuestas trazables con enlace a la fuente; asistencia proactiva que guía la exploración; memoria entre sesiones; baja latencia (respuesta en tiempo real). <i>Directos</i> .	Mayor valor percibido desde el primer uso y menor fricción de adopción.
Institución académica o de investigación	Acceso centralizado, colaborativo y actualizado frente al escritorio aislado (directo); colaboración y validación entre pares habilitadas por la plataforma (esperado).	Adopción a nivel de equipos y laboratorios; tracción hacia la masa crítica.
Empresa / plataforma (Molecular Craft)	Captación y retención como palanca potencial de masa crítica; captura de datos estructurados a partir de cada interacción con el asistente (proteínas consultadas, literatura y rutas de seguimiento), que habilita analítica de uso y modelos para captar nuevos usuarios —efectos de red generados por máquinas—; diferenciación competitiva. <i>Esperados</i> .	Aceleración hacia el umbral de masa crítica, base de datos para instrumentar la adopción y sostenibilidad del modelo de suscripción.

En concreto, los beneficios directos comprenden la trazabilidad bibliográfica, la continuidad conversacional y el apoyo a la exploración científica. Los esperados abarcan la mejora en la retención, el aprendizaje sobre patrones de uso y el fortalecimiento del modelo SaaS mediante datos estructurados. Estos últimos son proyecciones razonables, no resultados medidos: su validación exige instrumentación analítica y seguimiento del uso en producción, conforme al diseño de la Sección 3.6.

Esta lectura por *stakeholder* muestra que la propuesta de valor no se agota en el usuario final. El valor entregado al investigador desde el primer uso es la vía por la cual se activan los beneficios institucionales y, en última instancia, los de la plataforma como negocio. Es la lógica de masa crítica y efectos de red que enmarca el trabajo (Cervini & Joglekar, 2025).

4.11. Discusión de resultados

Los resultados expuestos muestran que la solución es técnicamente viable y coherente con los objetivos del trabajo. Su interpretación, sin embargo, debe situarse dentro del alcance exploratorio del estudio. El prototipo valida la factibilidad de integrar asistencia agéntica, memoria conversacional y recuperación de literatura verificable. No permite, en cambio, demostrar impactos concluyentes sobre la adopción, la retención o el desempeño comercial del modelo SaaS. Este apartado interpreta los resultados en tres planos: su relación con los objetivos, su contraste con el estado del arte y el efecto de las limitaciones sobre lo que puede afirmarse.

4.11.1. Lectura frente a los objetivos

Los cinco objetivos específicos se cumplieron, aunque no todos con el mismo grado de evidencia. Los cuatro primeros son objetivos de construcción: migración web, arquitectura agéntica, integración de fuentes externas y comportamiento proactivo. Su cumplimiento se verifica en el artefacto y se respalda en la suite de pruebas (Sección 4.6). El quinto es de otra naturaleza, pues exige mostrar que la asistencia agéntica puede operar como mecanismo de captación. Aquí la evidencia disponible es demostrativa, no experimental. La comparación entre el flujo de escritorio y el flujo web asistido ilustra la diferencia de experiencia, pero no la mide. Explicitar esta asimetría es preferible a presentar como resultado lo que todavía es una hipótesis fundada.

4.11.2. Contraste con el estado del arte

En el plano del diseño agéntico, el asistente instancia el bloque del LLM aumentado y combina dos de los patrones sistematizados por Anthropic: la paralelización y el enrutamiento (Anthropic, 2024). Se aparta de ellos en un punto. El enrutador no es un modelo de lenguaje, sino una regla determinista sobre señales lingüísticas y el estado de la memoria. La decisión preserva latencia y costo, y hace el ruteo trazable y depurable. A cambio, traslada la fragilidad a los fraseos ambiguos, que un clasificador basado en LLM manejaría con mayor flexibilidad. Se trata de un compromiso explícito entre simplicidad y robustez.

La segunda desviación respecto de la literatura afecta a la fundamentación de las respuestas. El trabajo comparte el objetivo de RAG —anclar la respuesta en fuentes externas—, pero no su mecanismo (Fan et al., 2024; Zhang & Zhang, 2025). En lugar de construir un índice vectorial de pasajes, el asistente invoca PubMed como una herramienta más del grafo. Para el usuario científico, esta elección privilegia la trazabilidad por diseño: cada publicación se verifica en su fuente. El costo es de cobertura, pues el asistente solo alcanza lo que PubMed indexa y lo que la consulta formulada logra recuperar. En un dominio donde la verificabilidad pesa más que la exhaustividad, el intercambio resulta razonable; en otros dominios podría no serlo.

Un tercer hallazgo se refiere a la modularidad. La arquitectura de un agente especializado por sección es coherente con la planificación agéntica modular descrita en la literatura (Webb et al., 2025). El desarrollo mostró, además, un efecto no previsto: la modularidad no solo



organiza el código, sino que *delimita el dominio de conocimiento* del agente. Al acotar su alcance a la consulta de proteínas y su literatura asociada, se reduce por construcción el espacio de respuestas fuera de dominio. Con ello disminuye, también por construcción, el riesgo de alucinación.

Por último, el marco de los efectos de red generados por máquinas ofrece la lectura estratégica de estos resultados (Cervini & Joglekar, 2025). El asistente produce el insumo de ese ciclo: datos estructurados sobre el uso. El ciclo, no obstante, aún no se cierra, pues falta la capa analítica que convierta esos datos en mejoras del producto. Los casos de referencia revisados en el marco teórico —Schrödinger y Benchling— muestran el punto de llegada de esa dinámica, no su punto de partida. Este trabajo se sitúa, con claridad, en el punto de partida.

4.11.3. Validez externa y alcance de la generalización

La validación se realizó sobre una única sección: la búsqueda en el Protein Data Bank. Su generalización al resto de la plataforma es plausible, pero descansa en dos supuestos. El primero es que cada sección disponga de una fuente externa verificable, análoga al par PDB/PubMed. El segundo es que sus tareas admitan descomponerse en nodos especializados acotados. Algunas secciones podrían carecer de una fuente autoritativa clara; la generación de ligandos es el caso más evidente. En esos escenarios, el patrón no se replicaría de forma directa y el mecanismo de fundamentación exigiría un rediseño.

4.11.4. Efecto de las limitaciones sobre lo que puede afirmarse

Las limitaciones metodológicas acotan con precisión el alcance de las afirmaciones. La ausencia de validación con usuarios impide sostener enunciados sobre usabilidad percibida, tiempo de tarea o satisfacción; la de telemetría, sobre continuidad de uso, conversión o retención. La comparación con el flujo de escritorio es cualitativa: ilustra una diferencia, pero no la cuantifica. Es coherente con lo que advierte la literatura sobre evaluación de agentes: las pruebas automatizadas verifican el comportamiento del sistema, pero no sustituyen la evaluación con usuarios en contexto real (Mohammadi et al., 2025). El Cuadro 8 delimita, dimensión por dimensión, qué quedó validado y qué no. En consecuencia, las conclusiones se sostienen en el plano de la viabilidad técnica y del valor funcional. Medir el impacto sobre la adopción es una tarea que este trabajo habilita más que zanja. El instrumento, eso sí, queda definido: el diseño de validación de la Sección 3.6 y las métricas del Cuadro 4.



5. Conclusiones

El trabajo desarrollado permite concluir que la incorporación de un asistente agéntico de bioinformática en un entorno web de Molecular Craft es técnicamente viable. Es, además, coherente con la evolución de la plataforma hacia un modelo SaaS. Su principal aporte no radica en automatizar consultas, sino en tres efectos concurrentes: aumenta el valor percibido desde el primer uso, mejora la trazabilidad de la interacción científica y abre la posibilidad de capturar datos estructurados sobre ella. El alcance del estudio es, con todo, exploratorio. La validación realizada demuestra factibilidad y valor funcional, pero no permite afirmar de forma concluyente el impacto del asistente sobre métricas reales de captación, permanencia o conversión. La siguiente etapa debe orientarse, por tanto, a la evaluación con usuarios reales y a la validación longitudinal de su aporte al modelo de negocio.

Los cinco objetivos específicos se cumplieron. Se migró a un entorno web la sección de búsqueda en el Protein Data Bank (OE1). Se diseñó e implementó la arquitectura agéntica como un grafo de estados con nodos especializados —resolución de proteínas, búsqueda bibliográfica y agregación— y memoria conversacional persistente sobre PostgreSQL (OE2). Se integraron fuentes científicas externas mediante la API del Protein Data Bank y un servidor MCP de PubMed, de modo que cada respuesta se ancla en publicaciones reales enlazadas a su origen (OE3). Se implementó el comportamiento proactivo, que reutiliza la memoria de publicaciones, aplica un mecanismo de respaldo y propone rutas de continuación (OE4). Y se contrastó, mediante una demostración comparativa, el flujo de escritorio con el flujo web asistido (OE5). El detalle de cada uno se documenta en el Capítulo 4.

La interpretación crítica de estos resultados se desarrolla en la Sección 4.11. Allí se contrastan con el estado del arte y se explicitan los compromisos de diseño asumidos: el ruteo determinista frente a un clasificador basado en LLM, y la invocación de herramientas frente a una arquitectura RAG. Se discuten, asimismo, los supuestos que condicionan su generalización al resto de la plataforma.

Entre las **fortalezas** de la solución destaca la trazabilidad de la información. A diferencia de los asistentes que devuelven referencias plausibles pero no siempre comprobables, este asistente entrega publicaciones directamente verificables en su fuente. A ello se suman dos rasgos. El primero es una experiencia de uso reforzada por el carácter anticipatorio del asistente y por la baja latencia del modelo empleado. El segundo es una arquitectura modular, que habilita la réplica del enfoque en otros módulos de la plataforma.

Las **limitaciones** del trabajo son igualmente explícitas. Se trata de un piloto sobre una sola sección, que no reimplementa la totalidad de la aplicación de escritorio. Falta, asimismo, una medición formal del *engagement* que permita cuantificar el efecto sobre la captación de usuarios. Reconocerlas acota el alcance de lo afirmado y, a la vez, ordena las líneas de continuación.

Con todo, la contribución estratégica más relevante de este trabajo es haber abordado, por diseño, una carencia estructural del modelo de escritorio planteada al inicio: la imposibilidad de capturar información estructurada sobre el uso. Cada interacción con el asistente genera ahora datos estructurados —proteínas consultadas, literatura recuperada, preguntas de seguimiento y rutas de continuación aceptadas—. Esos datos son la materia prima para reconocer patrones



de uso, inferir intereses y áreas de investigación dominantes, y construir modelos orientados a captar y retener usuarios. Conviene precisar, eso sí, que convertirlos en analítica accionable exige todavía la capa de instrumentación —identidad de usuario, telemetría y observabilidad— de la que el prototipo carece. Más que demostrar por sí mismo un aumento de la adopción, el asistente instala el mecanismo que permite, en trabajo futuro, medirla e impulsarla.

Como **trabajo futuro**, y en orden de prioridad, se proponen cuatro direcciones. La primera es condición para sostener empíricamente la tesis central de captación: ejecutar el diseño de validación definido en la Sección 3.6, instrumentando las métricas del Cuadro 4 y contrastando la propuesta con un grupo de investigadores reales. La segunda es replicar el asistente en otras secciones de la plataforma, como la búsqueda en ChEMBL o el visualizador 3D, lo que a la vez pondría a prueba los supuestos de generalización discutidos. La tercera es enriquecer sus capacidades, incorporando un segundo agente con responsabilidades complementarias o una búsqueda web profunda que amplíe las fuentes. La cuarta, en el plano estratégico, es profundizar el análisis del modelo de negocio SaaS para Molecular Craft, evaluando su viabilidad integral frente al modelo actual.

En síntesis, la principal contribución de este trabajo no consiste en demostrar de manera concluyente un aumento de la captación o la retención de usuarios. Consiste en validar un mecanismo técnicamente viable y estratégicamente pertinente, que habilita dos cosas a la vez: una mejora de la experiencia desde la primera interacción y la generación de datos estructurados para futuras decisiones de producto. Sobre esa base, concreta y replicable, puede articularse en las etapas siguientes el papel de la asistencia inteligente como palanca de adopción en la evolución de Molecular Craft hacia un modelo en la nube (Cervini & Joglekar, 2025).



Anexos

Esta sección reúne material complementario al desarrollo: capturas del flujo de uso del asistente sobre la sección de búsqueda en el Protein Data Bank.

Anexo A: Capturas del flujo del asistente

Las Figuras 12 y 13 ilustran el flujo de uso del asistente sobre la sección de búsqueda en el Protein Data Bank.

CONVERSACIÓN

+ Nueva conversacion

Dame un resumen completo sobre la proteína 7K9R

La proteína 7K9R corresponde a una proteína similar a Hsp90, conocida como TRAP1 (TNF Receptor Associated Protein 1), una chaperona molecular mitocondrial. TRAP1 desempeña un papel crucial en la regulación de la función mitocondrial, el metabolismo celular y la respuesta al estrés, siendo de particular interés por su implicación en diversas patologías, especialmente en el cáncer, donde actúa como un factor pro-tumoral.

Los datos estructurales disponibles para 7K9R son los siguientes:

PDB ID	Nombre	Organismo	Resolución	Método Experimental
7K9R	Hsp90-like protein (TRAP1)	Humano	1.95 Å	Difracción de Rayos X

Figura 12: Selección de proteína y resumen generado por el asistente.



Artículos de PubMed para "Hsp90-like protein"

Título	Autores	Año	DOI
New allosteric modulators of molecular chaperone TRAP1 from the integration of computational biology, medicinal chemistry, and biophysics.	Guarra F, Komarov D, Ciamarone A et al.	2026	10.1016/j.cstres.2026.100162
Inhibition of TRAP1 in the C-terminal domain influences mitochondria properties and breast cancer cell metabolism.	Mathieu C, Branco JR, Zancan P et al.	2026	10.1016/j.biopha.2026.119060
Cancer suppresses mitochondrial chaperone activity in macrophages to drive immune evasion.	Zhao H, Park J, Wang Y et al.	2025	10.1038/s41590-025-02324-2
TRAP1 and its therapeutic potential.	Gutierrez A, Archdeacon J, Blagg BSJ.	2025	10.1016/j.bmcl.2025.130395
The molecular chaperone TRAP1 promotes translation of Luc7I3 mRNA to enhance ovarian cancer cell proliferation.	De Lella S, Pedalino L, Abaker MAA et al.	2025	10.1261/rna.080588.125
TRAP1 expression elicits pro-tumoral functions in macrophages associated to malignant peripheral nerve sheath tumor cells.	Scantamburlo F, Rubini A, Toffanin M et al.	2025	10.1186/s13046-025-03525-1
TRAP1 promotes self-renewal and tumorigenicity of glioblastoma stem cells by inhibiting Notch1 ubiquitination.	Feng M, Zhang X, Zhou Y.	2025	10.1016/j.bbrc.2025.152528
Restricting metabolic plasticity enhances stress adaptation through the modulation of PDH and HIF1A in TRAP1-depleted colon cancer.	Tsai HY, Chen MH, Yun J et al.	2025	10.1016/j.canlet.2025.217977
Disulfide-mediated tetramerization of TRAP1 fosters its antioxidant and pro-neoplastic activities.	Faienza F, Laquatra C, Castelli M et al.	2025	10.1016/j.redox.2025.103677
Mitochondrial HSP90 Paralog TRAP1 Deletion Drives Glutamine Addiction in Tumor Cells via Destablization of the Cys/Glu Antiporter SLC7A11/xCT.	Joshi A(#), Dai L(#), Maisiak M et al.	2025	10.1158/1541-7786.MCR-24-0194

¿Quieres explorar más?

¿Cuáles son los inhibidores conocidos de Hsp90?

🔍 [Buscar otras chaperonas de "Mycobacterium tuberculosis"](#)

¿Qué diferencias estructurales tiene 7K9R con la Hsp90 humana?

Figura 13: Tabla de publicaciones con enlaces y sugerencias proactivas del asistente.



Referencias

- Anthropic. (2024, diciembre). Building effective agents. <https://www.anthropic.com/engineering/building-effective-agents>
- Artificial Analysis. (2026). Gemini 2.5 Flash: Intelligence, performance & price analysis. <https://artificialanalysis.ai/models/gemini-2-5-flash>
- Asplund, J.-E. (2024). Benchling revenue, valuation & funding. *Sacra*. <https://sacra.com/c/benchling/>
- Cao, C., Zheng, M., Huang, S., & Shao, X. (2026). How do subscription-based products hook users? SaaS user satisfaction and continued purchasing from information ecology perspective. *Industrial Management & Data Systems*, 126(3), 744-784. <https://doi.org/10.1108/IMDS-05-2024-0436>
- Cervini, P., & Joglekar, N. (2025). Machine-generated network effects: Implications for data, AI, and platform governance. *Management and Business Review*, 5(5). <https://doi.org/10.1177/2694104X251359812>
- Clifford, M. (2018). Validated SaaS solutions for drug development. *IDBS*. <https://www.drugdiscoverytrends.com/validated-saas-solutions-for-drug-development/>
- CrewAI. (2025). CrewAI documentation. <https://docs.crewai.com/>
- Fan, W., Ding, Y., Ning, L., Wang, S., Li, H., Yin, D., Chua, T.-S., & Li, Q. (2024). A survey on RAG meeting LLMs: Towards retrieval-augmented large language models. *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '24)*, 6491-6501. <https://doi.org/10.1145/3637528.3671470>
- Grand View Research. (2025). Artificial intelligence in drug discovery market size, share & trends analysis report. <https://www.grandviewresearch.com/industry-analysis/artificial-intelligence-drug-discovery-market>
- Guo, T., Chen, X., Wang, Y., Chang, R., Pei, S., Chawla, N. V., Wiest, O., & Zhang, X. (2024). Large language model based multi-agents: A survey of progress and challenges. *Proceedings of the 33rd International Joint Conference on Artificial Intelligence (IJCAI-24), Survey Track*, 8048-8057. <https://doi.org/10.24963/ijcai.2024/890>
- Katsamakas, E., & Sánchez-Cartas, J. M. (2025). Congestion, network effects and platform competition. *Journal of Economic Interaction and Coordination*, 20(2), 451-475. <https://doi.org/10.1007/s11403-024-00433-z>
- LangChain. (2025). LangGraph documentation. <https://langchain-ai.github.io/langgraph/>
- Lewis, J. R. (2018). The System Usability Scale: Past, present, and future. *International Journal of Human-Computer Interaction*, 34(7), 577-590. <https://doi.org/10.1080/10447318.2018.1455307>
- Liu, Y., Zhang, G., Wang, K., Li, S., & Pan, S. (2025). Graph-augmented large language model agents: Current progress and future prospects [Preprint en arXiv:2507.21407]. <https://arxiv.org/abs/2507.21407>
- Mangul, S., Martin, L. S., Eskin, E., & Blekhman, R. (2019). Improving the usability and archival stability of bioinformatics software. *Genome Biology*, 20, 47. <https://doi.org/10.1186/s13059-019-1649-8>



- Masterman, T., Besen, S., Sawtell, M., & Chao, A. (2024). The landscape of emerging AI agent architectures for reasoning, planning, and tool calling: A survey [Preprint en arXiv:2404.11584]. <https://arxiv.org/abs/2404.11584>
- Medeiros, D., Schieffer, G., Wahlgren, J., & Peng, I. (2023). A GPU-accelerated molecular docking workflow with Kubernetes and Apache Airflow. En *High Performance Computing. ISC High Performance 2023 International Workshops* (pp. 193-206, Vol. 13999). Springer. https://doi.org/10.1007/978-3-031-40843-4_15
- Microsoft. (2025). AutoGen documentation. <https://microsoft.github.io/autogen/>
- Model Context Protocol. (2025, noviembre). Model Context Protocol (MCP) specification, version 2025-11-25. Consultado el 14 de julio de 2026, desde <https://modelcontextprotocol.io/specification/2025-11-25>
- Model Context Protocol Team. (2025). One year of MCP: November 2025 spec release. <https://blog.modelcontextprotocol.io/posts/2025-11-25-first-mcp-anniversary/>
- Mohammadi, M., Li, Y., Lo, J., & Yip, W. (2025). Evaluation and benchmarking of LLM agents: A survey. *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '25)*. <https://doi.org/10.1145/3711896.3736570>
- Molecular Craft. (2025). Acerca de — Molecular Craft. <https://molecular-craft.feriadesoftware.cl/services.html>
- Patel, H. V., Jarvik, G. P., Hall, T. O., Veenstra, D. L., Xie, S. J., & Crosslin, D. R. (2026). Using user-centered design to better understand challenges faced during genetic analyses by novice genomic researchers. *Frontiers in Bioinformatics*, 6, 1849524. <https://doi.org/10.3389/fbinf.2026.1849524>
- Pratap, A. (2024). The rise of subscription-based AI platforms for drug discovery. *Chemical & Engineering News (C&EN)*. <https://cen.acs.org/business/informatics/rise-subscription-based-AI-platforms/102/web/2024/03>
- Precedence Research. (2025). Artificial intelligence (AI) in drug discovery market size, share and trends. <https://www.precedenceresearch.com/artificial-intelligence-in-drug-discovery-market>
- Qu, C., Dai, S., Wei, X., Cai, H., Wang, S., Yin, D., Xu, J., & Wen, J.-R. (2025). Tool learning with large language models: A survey. *Frontiers of Computer Science*, 19(8), 198343. <https://doi.org/10.1007/s11704-024-40678-2>
- Rusinko, A., Rezaei, M., Friedrich, L., Buchstaller, H.-P., Kuhn, D., & Ghogare, A. (2024). AIDDISON: Empowering drug discovery with AI/ML and CADD tools in a secure, web-based SaaS platform. *Journal of Chemical Information and Modeling*, 64(1), 3-8. <https://doi.org/10.1021/acs.jcim.3c01016>
- Schrödinger, Inc. (2025). Schrödinger reports strong fourth quarter and full-year 2024 financial results. <https://ir.schrodinger.com/press-releases/news-details/2025/Schrödinger-Reports-Strong-Fourth-Quarter-and-Full-Year-2024-Financial-Results/default.aspx>
- Shapouri, F., Ward, K., & Setor, T. (2024). Determinants of software as a service (SaaS) adoption. *Journal of Computer Information Systems*, 64(2), 301-313. <https://doi.org/10.1080/08874417.2023.2199270>
- Wang, L., Ma, C., Feng, X., Zhang, Z., Yang, H., Zhang, J., Chen, Z., Tang, J., Chen, X., Lin, Y., Zhao, W. X., Wei, Z., & Wen, J.-R. (2024). A survey on large language



- model based autonomous agents. *Frontiers of Computer Science*, 18(6), 186345. <https://doi.org/10.1007/s11704-024-40231-1>
- Watson, P., Hiden, H., & Woodman, S. (2013). Developing cloud applications using the e-Science Central platform. *Philosophical Transactions of the Royal Society A*, 371(1983). <https://doi.org/10.1098/rsta.2012.0085>
- Webb, T., Mondal, S. S., & Momennejad, I. (2025). A brain-inspired agentic architecture to improve planning with LLMs. *Nature Communications*, 16, 8633. <https://doi.org/10.1038/s41467-025-63804-5>
- Xi, Z., Chen, W., Guo, X., He, W., Ding, Y., Hong, B., Zhang, M., Wang, J., Jin, S., Zhou, E., Zheng, R., Fan, X., Wang, X., Xiong, L., Zhou, Y., Wang, W., Jiang, C., Zou, Y., Liu, X., ... Gui, T. (2025). The rise and potential of large language model based agents: A survey. *Science China Information Sciences*, 68(2), 121101. <https://doi.org/10.1007/s11432-024-4222-0>
- Zhang, W., & Zhang, J. (2025). Hallucination mitigation for retrieval-augmented large language models: A review. *Mathematics*, 13(5), 856. <https://doi.org/10.3390/math13050856>