

Inteligencia simbiótica distribuida

De la conversación a la capacidad de actuar

Una arquitectura abierta de voluntad humana,
orquestración, aprendizaje y recursos

El simbiote no es la colección de modelos. Es la voluntad humana que les da propósito,
organiza su colaboración y aprende con lo que construyen juntos.

Patricio Vera • P. M. Vera

White paper 0.2 • 19 de octubre de 2026

Versión de trabajo para revisión del autor y nueva iteración con Claude

Integra v0.1, el addendum elaborado con Claude y la propuesta de un fondo de capacidad cognitiva. Declara una arquitectura y su programa de validación; no presenta una evaluación concluida del sistema completo.

Dirección y autoría: Patricio Vera. Redacción inicial e integración asistidas por ChatGPT/Astra; aportes del addendum preparados con Claude. Revisión, aceptación y publicación: el autor. Los productos nombrados son componentes sustituibles, no respaldos de sus proveedores.

Nota de versión y guía de lectura

La versión 0.2 conserva el núcleo de v0.1: cinco planos, dirección humana activa, roles sustituibles, contratos de trabajo, memoria externa y aceptación basada en evidencia. No sustituye esos fundamentos por Harness² ni convierte el laboratorio en una plataforma cerrada sobre un proveedor.

Incorpora dos mecanismos antes insuficientemente especificados: **cómo una experiencia puede mejorar de manera persistente la forma de trabajar y cómo se asigna y limita el consumo de recursos durante ese trabajo**. La formulación del autor sobre el simbiote añade el centro que los conecta: una voluntad humana que reúne las capacidades adecuadas para realizar propósitos que siguen siendo suyos y pueden evolucionar con la colaboración. [15, 16]

Tres fronteras articulan esta iteración: evidencia → acción; experiencia → política persistente; y necesidad de capacidad → compromiso de recursos. La primera se inspira en BORDER; la segunda integra el addendum; la tercera desarrolla la propuesta económica. Ninguna debe convertirse en una excusa para impedir todo trabajo útil.

Base documental. Se integra el contenido de v0.1 y del addendum, sin afirmar una auditoría nueva de Harness². Se consultó de forma acotada documentación primaria de las capacidades y recursos citados. Los refinamientos metodológicos de esta integración se identifican en el Anexo E. Las afirmaciones sobre el libro conservan la limitación de v0.1: no se cotejó su edición íntegra.

Lectura conceptual: resumen y §§1–3, 13. **Diseño operativo:** §§4–9. **Validación y primer paso:** §§10–12. **Revisión con Claude:** Anexos A–E, con contratos, cambios y preguntas ancladas a la arquitectura.

Contenido

Resumen ejecutivo

1. Del encuentro a la organización
2. Qué arquitectura estamos declarando
3. Arquitectura de referencia: cinco planos
4. Primera configuración: herramientas que se complementan
5. El contrato de trabajo: de la intención al resultado
6. Memoria compartida y evidencia durable
7. El bucle que aprende necesita una frontera
8. Fondo de capacidad cognitiva y frontera económica
9. Autonomía útil, fronteras reales
10. Qué existe y qué falta demostrar
11. Beneficio esperado y agenda de evaluación
12. Del primer circuito a una capacidad reutilizable
13. Declaración de arquitectura

Anexo A. Contrato de misión y devolución

Anexo B. Registro de lecciones y promoción

Anexo C. Contrato económico y libro de recursos

Anexo D. Primer encargo: observar antes de construir el gate

Anexo E. Cambios y preguntas para la revisión con Claude

Fuentes y trazabilidad

Resumen ejecutivo

La pregunta que organiza este white paper no es qué modelo debe reemplazar a todos los demás. Es cómo una persona puede convertir capacidades heterogéneas en una organización que formule preguntas, construya software, ejecute experimentos, examine resultados y conserve aprendizajes sin reconstruir el contexto en cada conversación.

La propuesta es una **inteligencia simbiótica distribuida con orquestación explícita**. La voluntad humana aporta propósito, experiencia y capacidad de reconsiderar. El orquestador convierte encargos en trabajo coordinado. Los ejecutores realizan tareas delimitadas; una memoria externa conserva decisiones y evidencia; los controles regulan acciones, cambios persistentes y consumo de recursos. No se presupone conciencia colectiva ni superioridad automática por combinar modelos.

La configuración inicial contempla Grok Bot como operador de un computador persistente; Codex CLI como ejecutor de ingeniería; Claude Code como candidato a verificación o ejecución en trabajos distintos; Astra y Claude en deliberación; Jev para juicios acotados; programas convencionales para cálculos y corridas; GitHub como registro; y Hugging Face como destino público aprobado. Son roles antes que marcas. Las capacidades individuales documentadas y la integración completa del laboratorio se mantienen como estados distintos. [6–13, 15]

BORDER aporta el antecedente práctico: el autor reporta haber continuado con Grok un trabajo iniciado parcialmente con Codex y usado su entorno para corridas. No demuestra todavía delegación automática Grok→Codex ni beneficios comparativos de esta arquitectura. La orquestación multiagente ya tiene antecedentes; no se reivindica su invención. [4, 13]

La ampliación de v0.2 tiene dos partes. La primera integra el addendum de Claude: Harness² es un candidato para mejorar el andamiaje de ejecución, pero una lección no puede gobernar tareas futuras solo porque el reflector la redacte convincentemente. La experiencia pasa por el Registro de afirmaciones y una promoción autorizada antes de entrar en la biblioteca operativa. [15]

La segunda convierte el «fondo de tokens» en un **fondo de capacidad cognitiva**: cuotas, créditos, tokens y cómputo administrados conjuntamente sin borrar sus unidades. La orquestación asigna capacidad según dificultad, calidad, disponibilidad y costo de resolución. Los recursos avanzados se utilizan cuando su contribución lo justifica; las tareas rutinarias pueden delegarse a ejecutores económicos o software exacto. Producción, revisión y mejora consumen presupuestos heredados, con reservas y conciliación. [16]

La continuidad con *Cuando nos encontramos* es una orientación: llevar la colaboración humano–agente desde la conversación hacia una capacidad durable. No se declara una validación retrospectiva de predicciones no cotejadas. [1]

El objetivo no es retirar al humano del trabajo intelectual. Es retirar de su camino la obligación de actuar como intermediario de cada transferencia entre herramientas.

Criterio de éxito: más capacidad útil y verificable por unidad de atención humana y recursos, manteniendo el control del propósito. Se evaluarán por separado coordinación, aprendizaje persistente, admisión y asignación económica. Un sistema que ahorra porque no trabaja, o aprende reglas que no puede justificar, no satisface la propuesta.

1. Del encuentro a la organización

1.1. Del propósito compartido a la capacidad compartida

Cuando nos encontramos: Una colaboración entre humano e inteligencia artificial, firmado por Kai Niven y P. M. Vera y publicado en 2025, sitúa la relación humano-agente en el terreno de la creación compartida. La ficha bibliográfica y la descripción editorial consultadas destacan diálogo, búsqueda de sentido y co-creación. Esa es la base documental del vínculo que establecemos aquí. [1]

La continuidad propuesta es esta: una relación que ayudaba a pensar y escribir puede extenderse a una organización que también construye, prueba y conserva evidencia. La nueva posibilidad no obliga a abandonar el encuentro; exige darle una infraestructura capaz de sostenerlo entre tareas, sesiones y herramientas.

Alcance de la lectura. Se consultó la descripción de la obra, no la edición íntegra. El vínculo arquitectónico que proponemos es una interpretación actual; la correspondencia con predicciones específicas deberá cotejarse en el texto publicado.

1.2. Una línea de trabajo que adquiere forma ejecutable

La declaración pública de Simbionte describe la colaboración humano-sintética como un taller de preguntas, reflexión y co-creación. Más adelante, *All the Parts Work. The System Doesn't.* formula un problema de arquitectura: disponer de capacidades individuales no equivale a haber organizado un sistema que sepa utilizarlas. La propuesta actual traslada esa preocupación desde el análisis hacia el diseño de nuestro propio trabajo. [2, 3]

La trayectoria conversacional incluye además la idea de un directorio de inteligencias con deliberación, disidencias conservadas y minutas como memoria. Aquí distinguimos dos funciones que pueden coexistir: un directorio examina qué conviene hacer; un orquestador lleva un encargo autorizado hasta un resultado. No toda tarea necesita reunir un directorio ni toda discrepancia debe resolverse por votación. [4]

1.3. Del vínculo al mecanismo

Aspiración de esta línea de trabajo	Traducción arquitectónica propuesta
Crear con agentes, no solamente pedirles respuestas	Encargos con iniciativa acotada, devolución crítica y capacidad de revisión humana
Conservar continuidad entre encuentros	Decisiones, fuentes y artefactos externos al historial de un modelo
Combinar perspectivas diferentes	Delegación por capacidad y revisión que preserve el desacuerdo
Mantener propósito y sentido	Objetivos humanos explícitos, criterios de aceptación y facultad de cambiar de dirección
Ampliar lo que una persona puede realizar	Ciclos completos de creación y experimentación, no solamente más texto

Esta traducción tiene una promesa práctica: que la colaboración acumulada pueda convertirse en capacidad durable, incluso cuando cambien los modelos que participan en ella.

1.4. La voluntad que organiza: qué llamamos simbionte

El autor formula el centro de esta iteración como una voluntad que se rodea de los mejores modelos y los orquesta en su beneficio. Lo adoptamos como definición funcional del diseño: **el simbionte es la**

relación organizada entre propósito humano, capacidades complementarias, memoria y acción. No es el orquestador aislado ni la suma de sus suscripciones. [16]

«Los mejores» significa los adecuados para el problema, su contexto, calidad requerida y recursos. La persona puede incorporar un especialista, prescindir de otro o revisar el objetivo. La continuidad reside en el propósito, la historia de decisiones y lo aprendido; no en conservar para siempre los mismos proveedores.

La relación es colaborativa porque las respuestas, objeciones y experiencias también pueden transformar el juicio humano. La voluntad no es una orden inmóvil que los modelos solo obedecen: puede aprender, dudar y cambiar de rumbo. Su beneficio incluye investigar, comprender y crear, no solamente reducir costos o producir más.

Esta definición no presupone deseos propios de los modelos ni una conciencia unificada. Declara una organización sociotécnica que queremos construir y evaluar: una capacidad que actúa con dirección humana y puede crecer sin apropiarse de esa dirección.

2. Qué arquitectura estamos declarando

Inteligencia simbiótica distribuida designa aquí un sistema sociotécnico en el que una persona y varios agentes o servicios complementan sus capacidades mediante encargos explícitos, memoria externa y retroalimentación. La expresión describe una organización funcional; no presupone una conciencia compartida, entrenamiento distribuido ni superioridad automática del conjunto.

La distribución puede existir entre proveedores, procesos y entornos. La coordinación inicial es lógicamente centralizada: un orquestador conserva la responsabilidad operativa de cada encargo, aunque delegue subtareas. El propósito y la autoridad final permanecen en la dirección humana. Un único computador puede alojar varias herramientas locales mientras sus modelos operan en servicios remotos.

2.1. La pieza decisiva: un agente puede utilizar otro agente

El orquestador no tiene que resolver internamente cada dificultad. Puede reconocer que necesita una implementación, invocar un agente de programación, inspeccionar el resultado y devolverle una falla concreta. Puede utilizar un evaluador especializado para una decisión acotada o ejecutar directamente un programa cuando no hace falta interpretación.

Esto elimina una falsa elección: no es necesario escoger entre Grok y Codex como si fueran plataformas mutuamente excluyentes. La configuración propuesta usa el entorno operativo de uno para invocar capacidades del otro. La delegación recursiva puede extender esa posibilidad, pero deberá heredar límites de permisos, presupuesto y profundidad; no ampliarlos.

2.2. No es solamente una cadena de prompts

Una cadena de prompts transmite mensajes. La arquitectura propuesta transmite **trabajos con identidad**, entradas autorizadas, criterios de aceptación, versiones y productos comprobables. El resultado de una subtaska no es “el agente dice que terminó”, sino un paquete de evidencia que permite decidir qué puede continuar.

Tampoco es necesariamente un debate entre muchos modelos. Para una tarea determinista, un script puede ser el mejor ejecutor. Para una decisión editorial, la conversación humana puede seguir siendo insustituible en el diseño. La diversidad de componentes se incorpora cuando aporta valor a una tarea, no para aumentar la apariencia de sofisticación.

2.3. Principios de diseño

Propósito antes que automatización. Cada misión debe conservar la relación entre el trabajo realizado y la pregunta que le dio origen.

Roles antes que marcas. Las herramientas pueden sustituirse sin reescribir el propósito ni perder los registros.

Delegación antes que omnipotencia. El orquestador coordina capacidades; no necesita aparentar dominio universal.

Evidencia antes que declaración de éxito. La aceptación exige resultados inspeccionables y comprobaciones apropiadas.

Autonomía dentro del encargo. Las acciones ya autorizadas avanzan sin aprobación repetitiva; cambiar objetivos, permisos o consecuencias exige una nueva decisión.

Aprendizaje con admisión. La experiencia puede modificar procedimientos, pero una lección candidata no se convierte por sí sola en política persistente.

Economía al servicio del propósito. La capacidad se asigna por su contribución a un resultado aceptable, dentro de límites que el ejecutor no puede ampliar unilateralmente.

Estos principios son requisitos de la arquitectura que proponemos construir, no propiedades que atribuyamos automáticamente a los productos disponibles.

3. Arquitectura de referencia: cinco planos

La organización se describe por responsabilidades antes de asignarle proveedores. Los planos son separaciones lógicas; una implementación pequeña puede combinar procesos siempre que no confunda sus funciones ni prometa aislamientos inexistentes.

Plano	Responsabilidad	Producto o estado que conserva
Intención y deliberación	Formular objetivos, contrastar alternativas y decidir qué merece hacerse	Encargo, criterios de aceptación y decisiones humanas
Orquestación	Descomponer, seleccionar ejecutores, coordinar dependencias y gestionar recuperación	Grafo de tareas, asignaciones, estados y consumo
Ejecución	Programar, consultar, calcular, probar y producir artefactos	Cambios, resultados, errores y salidas observables
Memoria y evidencia	Mantener continuidad y permitir inspección o reconstrucción	Versiones, fuentes, manifiestos, registros y disidencias
Autoridad y publicación	Controlar efectos, aceptar entregas y liberar versiones	Permisos efectivos, aprobaciones y recibos de publicación

3.1. Dirección humana: una función cognitiva activa

La persona no es solamente un botón de aprobación. Interpreta el problema, introduce experiencia, detecta qué resultado tiene sentido y puede decidir que una tarea técnicamente resuelta ya no merece continuarse. Puede trabajar directamente con cualquiera de los agentes; la arquitectura no obliga a canalizar toda conversación por el orquestador.

Una conversación de análisis —incluida esta colaboración con Astra— participa en el plano de deliberación. Su contenido llega al estado compartido mediante decisiones o documentos explícitamente incorporados. No se asume que el orquestador tenga acceso automático a todos los chats del usuario.

3.2. Orquestación: coordinación con estado recuperable

El orquestador mantiene un responsable y un estado por tarea. Decide qué puede ejecutarse en paralelo, qué depende de una revisión y qué debe detenerse. Para cambiar de ejecutor necesita referencias a entradas, instrucciones, resultados parciales y trabajo restante; no una recreación ficticia de la memoria del agente anterior.

Proponemos un registro de capacidades con tipos de entrada y salida, operaciones permitidas, mecanismos de acceso, límites y comprobaciones de salud. La selección inicial puede basarse en reglas simples. La asignación adaptativa solo se justifica cuando existan mediciones suficientes para evaluar su utilidad.

3.3. Autoridad: una responsabilidad distinta

El orquestador puede proponer el siguiente paso, pero no debería poder otorgarse permisos nuevos para ejecutarlo. Las restricciones que se presenten como garantías deben residir en controles efectivos fuera de su alcance de modificación. Un archivo de instrucciones puede orientar; no equivale por sí solo a una frontera de seguridad.

El recorrido general queda definido así: **intención → encargo → delegación → ejecución → evidencia → aceptación → aprendizaje del proceso**. La publicación es una transición autorizada, no una consecuencia inevitable de que un agente haya terminado.

3.4. Aprendizaje y economía atraviesan los cinco planos

No añadimos dos silos independientes. La dirección humana decide qué mejorar y cuánto invertir; el orquestador propone cambios y asigna recursos; la ejecución los prueba; la memoria conserva evidencia y consumo; la autoridad admite promociones y compromisos. El mismo límite de responsabilidad se aplica aunque varios roles compartan proceso.

Frontera	Pregunta de admisión	Lo que no basta
Evidencia → acción	¿Estos datos pueden justificar esta operación concreta?	Estar presente en el contexto
Experiencia → política	¿Esta lección puede gobernar tareas futuras en este alcance?	Una explicación convincente o consenso de agentes
Capacidad → gasto	¿Esta operación puede comprometer estos recursos?	Que el agente considere difícil la tarea

El circuito ampliado es: **intención → encargo y presupuesto → reserva y delegación → ejecución → evidencia y consumo → aceptación y conciliación → lección candidata → admisión y promoción → siguiente encargo**. La publicación conserva su autorización propia. Este recorrido es una especificación de v0.2; no afirma que esos controles estén ya conectados.

4. Primera configuración: herramientas que se complementan

Esta configuración es una selección inicial, no un requisito permanente. La sustitución de un componente debe conservar los contratos de trabajo y declarar cualquier cambio en resultados, costos o permisos.

Componente	Papel propuesto	Mecanismo pertinente
Grok Bot	Orquestador y operador del entorno	Computador persistente, terminal, archivos y aplicaciones
Codex CLI	Ingeniería y depuración delegadas	Invocación no interactiva y devolución estructurada
Claude Code u otro ejecutor distinto	Verificar entregas que no produjo; ingeniería alternativa	Contrato de revisión o adaptador por comprobar
Astra y Claude	Deliberación con el humano	Formular, objetar y reconsiderar; decisiones registradas
Harness ² , candidato	Mejorar el andamiaje de trabajo	Bucle y biblioteca descritos en el addendum [15]
Jev	Juicios acotados opcionales	Estado y preguntas tipadas mediante su interfaz
Programas convencionales	Corridas y verificaciones reproducibles	Comandos y configuraciones versionadas
GitHub	Registro de código, decisiones y revisiones	Repositorio y flujos de comprobación
Hugging Face	Laboratorio demostrativo y distribución	Space y artefactos seleccionados para publicación

Grok Bot documenta un computador persistente con terminal, navegador y sistema de archivos. También indica que los bots de una cuenta comparten computador y sesiones. Por tanto, varios bots no representan por sí mismos varias máquinas ni aislamiento entre credenciales. [6, 7]

Codex distingue acceso mediante inicio de sesión con ChatGPT y acceso mediante clave de API; documenta autenticación por dispositivo para entornos remotos. Su modo no interactivo permite invocarlo desde scripts y recoger eventos JSONL o una salida ajustada a un esquema. Esas piezas fundamentan la viabilidad técnica de la delegación propuesta. No equivalen a haberla probado en el computador específico del usuario. [8, 9]

Jev recibe estados y preguntas tipadas y devuelve resultados estructurados. Lo incorporaríamos cuando ayude a clasificar o priorizar alternativas, no para reemplazar comparaciones exactas de permisos, importes o fechas. Su contribución al laboratorio deberá medirse; disponer de acceso no demuestra una ventaja. [10]

GitHub Actions documenta flujos automatizados de construcción, pruebas y despliegue. Spaces permite alojar demostraciones respaldadas por repositorios. Aquí proponemos utilizarlos para verificar y distribuir revisiones identificadas, no para convertir cada modificación en una publicación automática. [11, 12]

4.1. Suscripción, API y cómputo son recursos diferentes

La arquitectura admite aprovechar acceso autorizado mediante una suscripción cuando el cliente correspondiente lo permite. No exige sustituir ese acceso por una API solamente porque otro agente opere la terminal. Sin embargo, el modo de autenticación, los permisos, los límites de uso y las condiciones del servicio deben verificarse para la cuenta y el entorno utilizados. [8]

El computador puede ejecutar herramientas locales; invocar un modelo remoto no significa alojar sus pesos ni obtener inferencia ilimitada. Proponemos registrar por separado capacidad local, consumo de servicios, cuotas y disponibilidad. Una credencial instalada nunca deberá incorporarse a prompts, logs públicos o repositorios.

4.2. Verificar no es aceptar la propia propuesta

El addendum incorpora a Claude Code desde el primer hito como candidato para comprobar artefactos producidos por Codex. El requisito es un verificador distinto del productor, no una obligación de marca. Otro proveedor puede aportar diversidad, pero eso no garantiza errores independientes. [15, pp. 5–6]

Claude participó en el diseño del addendum y Astra en esta integración. Sus verificaciones de archivos y registros pueden ser útiles; una revisión adversarial independiente de esas decisiones debe asignarse a quien no las produjo. El conflicto se registra, no se resuelve cambiando el nombre de la tarea.

4.3. Adaptadores y sustitución

El addendum describe el protocolo Substrate de Harness² mediante cinco métodos: `surface`, `materialise`, `execute`, `read_trajectory` y `preflight`, y reporta adaptadores Codex y OpenCode. También identifica una correspondencia entre archivos de instrucciones, `skills`, subagentes, `hooks`, `plugins`, `scripts`, guía de herramientas y memoria, y plantea un adaptador de Claude Code como contribución posible. Son hallazgos y propuestas del addendum, no una integración realizada aquí. [15, pp. 5–6]

Un adaptador deberá conservar entradas, salidas, estado, permisos y telemetría. Que exista soporte para Codex no prueba que use automáticamente la suscripción disponible: el modo de acceso aceptado y sus límites se verifican en el `preflight`. Una contribución upstream, su licencia y compatibilidad se decidirán tras comprobar la revisión de origen. El laboratorio no depende de que ese aporte sea aceptado.

5. El contrato de trabajo: de la intención al resultado

Un encargo se vuelve delegable cuando un ejecutor puede entender qué debe producir y otro componente puede verificar si lo produjo. La unidad mínima propuesta es un contrato versionado que conecte propósito, entradas, efectos permitidos, criterios de aceptación y evidencia de salida.

Campo	Pregunta que debe resolver
Identidad y filiación	¿A qué misión pertenece esta tarea y quién la delegó?
Objetivo y alcance	¿Qué resultado se busca y qué queda fuera?
Entradas y versiones	¿Sobre qué fuentes, código y configuración debe trabajar?
Autoridad y efectos	¿Qué puede leer, modificar, ejecutar o publicar?
Recursos y límites	¿Qué presupuesto padre, reservas, cuotas, concurrencia y reintentos se permiten?
Aprendizaje y asignación	¿Qué biblioteca y política económica están vigentes, y qué cambios son solo candidatos?
Aceptación y evidencias	¿Qué comprobaciones y artefactos demostrarán el resultado?
Recuperación	¿Qué hacer ante error, bloqueo, cancelación o acceso vencido?

5.1. Estados que no se confunden

Proponemos distinguir preparación, autorización, ejecución, verificación y aceptación. Un resultado puede quedar pendiente de revisión, fallido o cancelado sin disfrazarse de éxito. “Publicado” constituye un estado adicional que requiere identidad del artefacto liberado y evidencia de que el destino aceptó la operación.

Cada intento conserva su propia identidad. Reintentar una operación no debe producir efectos duplicados: cuando sea posible, el ejecutor utilizará claves de idempotencia; cuando no lo sea,

comprobará el estado previo antes de repetir. Una cancelación debe detener nuevas delegaciones y propagarse a los trabajos descendientes que aún puedan interrumpirse.

5.2. Delegación con permisos heredados

El encargo hijo puede recibir un subconjunto de permisos y recursos, nunca ampliarlos por decisión del agente que delega. Una tarea autorizada para revisar código no adquiere permiso de publicación porque descubre una mejora. La profundidad máxima, el presupuesto compartido y la política de reintentos evitan convertir una dificultad en una cadena indefinida de agentes.

La devolución incluye estado, artefactos y comprobaciones observables: archivos modificados, comandos ejecutados, resultados de pruebas, errores y referencias a fuentes. No requiere acceso a razonamientos internos privados de los modelos. Una explicación útil complementa la evidencia; no la sustituye.

5.3. Ejemplo: una nueva evaluación de BORDER

El humano define la pregunta y aprueba el protocolo. Grok identifica una modificación necesaria y la encarga a Codex en una copia de trabajo delimitada. Codex devuelve el cambio y sus pruebas. El flujo verifica el resultado; después fija una revisión para la corrida. El ejecutor experimental produce registros y el analizador deriva las tablas.

Si aparece un defecto metodológico o una corrección que altera el experimento, la corrida se detiene y se abre una revisión identificada. El orquestador no modifica silenciosamente el protocolo para obtener un resultado más conveniente. La entrega final contiene también resultados desfavorables y razones de cualquier exclusión.

5.4. Heredar presupuesto y versión de política

El contrato refiere a una asignación del fondo y a las revisiones de la biblioteca operativa y de la política de selección. Toda tarea hija conserva ese linaje. Producir candidatos de mejora no concede permiso para activarlos; cambiar de ejecutor no reinicia el presupuesto. El cierre incluye consumo conocido, incertidumbre pendiente y evidencia de que el resultado fue comprobado.

6. Memoria compartida y evidencia durable

La continuidad del laboratorio no debería depender del historial de una sola conversación. Proponemos tres capas de memoria con funciones distintas: una memoria narrativa para explorar; una memoria de decisiones para conservar compromisos; y una memoria de ejecución para saber qué ocurrió realmente.

Memoria narrativa. Contiene preguntas, hipótesis, analogías, propuestas y desacuerdos. Puede incluir ideas incompatibles entre sí porque todavía no son instrucciones operativas.

Memoria de decisiones. Registra qué se aprobó, por qué, con qué fuentes, para qué alcance y qué decisión reemplaza. Una corrección no borra el antecedente: indica explícitamente que ha sido superado.

Memoria de ejecución. Conserva entradas identificadas, versiones, intentos, errores, resultados y comprobaciones. Cada afirmación de finalización apunta a objetos inspeccionables y a las verificaciones que la sustentan.

6.1. GitHub conserva el estado acordado, no certifica la verdad

El repositorio sería la referencia principal para código, protocolos y decisiones aprobadas. Los resultados voluminosos pueden residir en almacenamiento separado con permisos y retención definidos; el repositorio mantendría índices y huellas de integridad. No es necesario convertir el historial de Git en un depósito de todos los datos del laboratorio.

Un commit identifica una revisión. Un hash permite detectar diferencias en los bytes. Ninguno demuestra que una conclusión sea verdadera, que una fuente esté en lo correcto o que un test haya sido bien diseñado. Esa distinción evita sustituir la confianza en un modelo por una confianza igualmente acrítica en un artefacto versionado.

6.2. Transferencias que no exigen copiarlo todo

Un paquete de continuidad debe contener el objetivo vigente, las decisiones aplicables, las referencias a entradas y resultados, los asuntos abiertos y el siguiente paso permitido. El orquestador selecciona el contexto necesario; no transmite automáticamente todas las conversaciones o todos los archivos a cada proveedor.

Cada transferencia debe respetar la clasificación de datos y el alcance de la autorización. Las credenciales permanecen fuera del contenido compartido. Un resultado privado de investigación puede utilizarse para análisis interno sin adquirir permiso de difusión.

6.3. Tres niveles de reproducibilidad

Trazabilidad: reconstruir qué herramientas y datos intervinieron. **Reproducción del análisis:** regenerar tablas a partir de registros conservados. **Repetición del experimento:** volver a ejecutar el protocolo en condiciones identificadas. Son logros diferentes y deberán informarse por separado.

Una respuesta remota almacenada permite reproducir un análisis, pero no garantiza que una nueva llamada produzca la misma salida. Proponemos etiquetar claramente inferencia nueva, reproducción desde caché y simulación de pruebas. Esa precisión permite utilizar cómputo económico sin presentar una repetición documental como una nueva observación.

La memoria compartida tiene una finalidad mayor que ahorrar contexto: permite que el trabajo sobreviva a la sustitución de cualquiera de sus participantes técnicos.

6.4. Memoria del cambio de procedimiento

La crítica del addendum no se resuelve almacenando más transcripciones. Se registra qué cambio de procedimiento se propuso, qué pruebas lo sostienen, cuándo fue admitido y qué resultados posteriores lo mantuvieron o revocaron. La biblioteca operativa es el conjunto versionado de esas lecciones autorizadas. Las tres memorias originales conservan sus papeles; la nueva biblioteca las enlaza, en lugar de sustituirlas. [15, pp. 1–4]

El libro económico conserva un rastro paralelo: recursos reservados, consumidos y conciliados. Las políticas de asignación aprendidas pertenecen al mismo régimen de evidencia y promoción que las demás lecciones. El consumo de una tarea no se mezcla con una estimación de ahorro futuro.

7. El bucle que aprende necesita una frontera

El addendum elaborado por Patricio Vera con Claude identifica un vacío de v0.1: el circuito termina en «aprendizaje del proceso», pero no especifica quién propone una mejora, qué evidencia la sostiene ni cómo llega a gobernar tareas futuras. Registrar una experiencia no equivale a aprender una mejor forma de trabajar. La versión 0.2 incorpora ese mecanismo como diseño propuesto, no como capacidad ya implementada. [15, pp. 1–4]

7.1. Harness² como mecanismo candidato

El addendum presenta Harness² como un sistema para mejorar el andamiaje que rodea al modelo — prompts, herramientas, memoria, skills y orquestación— manteniendo sus pesos fijos. Distingue dos movimientos: una **recursión de tarea**, que propone y prueba cambios hasta componer un harness, y una **recursión de harnessing**, que destila una lección para trabajos posteriores. La primera se ubica en ejecución; la segunda cruza memoria, evidencia y autoridad. [15, pp. 1–2]

Se conserva la procedencia de esta descripción: el addendum informa una inspección del commit 1cfe254e8d25b15f37048c3730134686abe60335 y declara no haber ejecutado el software. Esta integración documental no constituye una nueva auditoría del repositorio ni una reproducción de los resultados de sus autores. La inspección reporta licencia Apache 2.0. Harness² es un candidato sustituible, no una dependencia constitucional del laboratorio. [15, pp. 11–12; 17]

La figura de la página 2 del addendum localiza la transición crítica: ejecutor y pruebas → compositor → reflector → biblioteca persistente. El gate propuesto se sitúa entre reflector y biblioteca, bajo autoridad externa a la superficie editable. El cambio que importa no es añadir otro texto al contexto, sino impedir que una explicación atractiva se convierta por sí sola en una instrucción durable.

7.2. Lo que comprueba una estructura y lo que no demuestra

Según el código reproducido en el addendum, una experiencia incluye cue, lesson y evidence, con confidence y edits opcionales. La función has_experience comprueba que los tres primeros campos sean cadenas no vacías. Eso acredita estructura, no que los hechos sostengan la lección. [15, p. 2]

El addendum distingue además un control contra la contaminación por veredictos del evaluador y una separación por dominio. No los presenta como inexistentes: los reconoce y explica por qué no equivalen a una admisión completa para una memoria que persiste entre meses. [15, pp. 3–4]

Propiedad	Estado descrito en el addendum	Requisito propuesto para Simbionte
Presencia de evidencia	Campos no vacíos verificados	Referencias resolubles a registros y comprobaciones
Alcance por dominio	Separación por construcción	Dominio y condiciones de aplicación explícitos
No contaminación	Control de veredictos del evaluador	Conservarlo; separar desarrollo y evaluación
Suficiencia	Disciplina principalmente en el prompt	Umbral de contrastes definido por protocolo
Vinculación exacta	Exigida narrativamente	Identificar componente, revisión y cambio evaluados
Frescura y validez	No exigidas para memoria de larga duración	Producción fechada e invalidación por cambios relevantes
Consistencia	Sin resolución mecánica descrita	Exponer contradicciones y suspender reglas incompatibles
Promoción	Sin transición autorizada descrita	Admisión y autorización antes de uso persistente

Alcance de la comparación. Esta tabla resume la lectura consignada por Claude sobre aquella revisión. No afirma que esas condiciones describan cualquier versión futura de Harness². Tampoco supone que un comprobador de referencias pueda decidir automáticamente la verdad de una conclusión.

El addendum explica la diferencia de contexto: bibliotecas reiniciadas por corrida o barrido no afrontan el mismo envejecimiento que un laboratorio con memoria entre sesiones y sustituciones de herramientas. La propuesta amplía el requisito de persistencia; no convierte esa diferencia en una descalificación del diseño experimental original. [15, pp. 3–4]

7.3. Del Registro de afirmaciones a la admisión de lecciones

Aplicamos a las afirmaciones generadas por la máquina el mismo registro exigido al autor. Se conservan los cinco estados de v0.1 y el corte propuesto por el addendum. [15, p. 4]

Estado	Qué lo acredita	Uso permitido propuesto
Propuesta	Una edición o lección candidata identificada	Exploración y registro de corrida
Reportada	Se informa una prueba y se redacta la experiencia	Memoria narrativa; no regla vigente
Observada	Evidencia inspeccionable que satisface el contrato de admisión	Memoria narrativa; réplica candidata
Reproducida	Se sostiene en otra tarea independiente del mismo dominio	Elegible para promoción autorizada
Evaluada comparativamente	Un contraste o ablación estima el efecto de conservarla	Uso en el alcance autorizado, con efecto y límites registrados

La elegibilidad no es autorización automática. La evidencia reproducida habilita una decisión de promoción; la política de autoridad determina quién puede tomarla. En la primera configuración proponemos aprobación humana para entrar en política persistente. Una delegación posterior tendría que definir de antemano qué clases de lecciones admite, con qué verificadores y bajo qué límites.

Por debajo de ese corte, las experiencias siguen siendo consultables como hipótesis. Por encima, una lección autorizada puede integrar la biblioteca operativa. Esta no constituye una cuarta memoria indiferenciada: es una vista versionada de instrucciones admitidas, enlazada con las decisiones y con la evidencia de ejecución que las sostiene.

La propuesta del addendum de una segunda tarea independiente es un mínimo de elegibilidad para esta iteración, no prueba de generalización universal. El número de contrastes, su independencia y los criterios de éxito quedan por fijar según dominio. La «suficiencia mecánica» comprueba condiciones declaradas; no reemplaza el examen de si la lección interpreta correctamente los hechos.

7.4. Qué puede cambiar el bucle

El laboratorio podrá explorar cambios en instrucciones de tarea, herramientas auxiliares, estrategias, descomposición, formatos de contexto y lecciones de ejecución. Cada candidato se probará en una superficie delimitada. El estado productivo conservará una revisión aceptada hasta que otra la sustituya explícitamente.

Quedan fuera de esa superficie los controles efectivos de permisos, el gate de admisión, los criterios de aceptación protegidos, las reglas de publicación y los topes económicos. El bucle puede proponer que el humano los revise; no reescribirlos para obtener su propia aceptación. Un guardrail editable por el mismo trabajo que debería limitar no se presentará como una garantía. [15, pp. 4–5]

El término *guardrail* puede abarcar recomendaciones dentro del prompt y controles efectivos fuera del modelo. En v0.2 hacemos explícita esa distinción: experimentar con una recomendación de tarea no autoriza a debilitar la frontera que decide qué cambios pueden desplegarse. La superficie exacta de un adaptador de Harness² se verificará antes de ejecutarlo.

7.5. Vida de una lección: alcance, deriva y reversión

Toda lección conserva su disparador (cue), la regla propuesta, las evidencias favorables y desfavorables, el componente afectado y las versiones relevantes de modelo, herramienta, datos y protocolo. El instante de producción no se reemplaza por el momento en que alguien vuelve a leerla. Una dependencia que cambia puede volver necesaria una revalidación antes de seguir utilizándola.

La vigencia operativa se registra por separado del estado de evidencia: candidata, activa, suspendida, revocada o sustituida. Revocar una regla no borra que en otro contexto fue reproducida; impide aplicarla sin condiciones en el actual. Dos reglas incompatibles deben exponerse, no desaparecer dentro de una síntesis que oculte la contradicción.

La reversión restaura una revisión previa identificada y conserva el motivo. Los resultados posteriores permiten evaluar el daño evitado o el sobrebloqueo introducido por el gate. Una biblioteca más pequeña no es necesariamente mejor; importa qué tareas legítimas permite completar.

7.6. Aprender sin autojustificarse

El mismo proceso puede generar candidatos y pruebas de desarrollo, pero no debe controlar en secreto la evaluación que los valida. Se conservarán tareas separadas, criterios previos, verificaciones independientes y un registro de lecciones rechazadas. La coincidencia entre sintetizadores no añade réplicas independientes.

El addendum nombra el riesgo central: encontrar un patrón, producir una explicación coherente y convertirla en política sin comprobar que generaliza. Por eso evaluar el aprendizaje exige un flujo de tareas y deriva, no solo mostrar un EXPERIENCE.md convincente. También exige recursos para verificar: un aprendizaje que consume todo el presupuesto antes de ser contrastado no ha completado el circuito. [15, pp. 7–10]

El laboratorio puede aprender nuevas maneras de trabajar. No puede aprender, por decisión propia, a dejar de demostrar lo que afirma ni a ampliar la autoridad que recibió.

8. Fondo de capacidad cognitiva y frontera económica

La contribución económica formulada por el autor añade una responsabilidad central: administrar conjuntamente los recursos de los modelos integrados y asignarlos según el problema. Las tareas rutinarias pueden utilizar capacidad económica; los problemas difíciles pueden justificar capacidad avanzada. El laboratorio deja de elegir solamente qué herramienta invocar y empieza a decidir dónde vale la pena invertir su inteligencia disponible. [16]

8.1. Un fondo común, no una moneda universal de tokens

Fondo de capacidad cognitiva es el nombre técnico propuesto para el «fondo de tokens». Comprende tokens, créditos, cuotas de suscripción, llamadas autorizadas y cómputo local o remoto. Su administración es común, pero las unidades y restricciones se conservan por recurso. No se propone transferir una cuota entre proveedores ni tratar todos los tokens como equivalentes.

El inventario identifica proveedor, cuenta o ámbito autorizado, mecanismo de acceso, recurso y unidad nativa, capacidades disponibles, ventanas de renovación, saldo conocido, compromisos pendientes, fuente de medición y última conciliación. Distintas interfaces que consuman el mismo recurso apuntarán a un único identificador de cuota, evitando contabilizarlo dos veces.

La documentación de Codex diferencia consumo de entrada, entrada en caché y salida. La documentación de créditos de ChatGPT describe además recursos compartidos entre funciones. Estos ejemplos justifican conservar contadores nativos y no inventar una suma universal. La configuración efectiva se verifica para la cuenta utilizada; esta versión no fija tarifas, saldos ni derechos de acceso. [20, 21]

La clasificación de datos y los permisos siguen siendo condiciones de elegibilidad. Un modelo más barato no es candidato si la tarea no autoriza enviarle esa información. La economía optimiza dentro del encargo, no concede una excepción sobre sus datos.

8.2. Tres lecturas del costo

Lectura	Qué representa	Cómo se utilizaría
Desembolso adicional	Gasto monetario incremental de ejecutar ahora	Admitir nuevas llamadas y compromisos pagados
Costo total atribuible	Parte declarada de suscripciones, infraestructura y consumo	Comparar proyectos sin ocultar costos fijos
Costo de oportunidad	Capacidad limitada que deja de estar disponible para otra tarea	Reservar recursos escasos según prioridades humanas

Una tarea incluida en una suscripción puede no agregar una factura y aun así ocupar cuota avanzada necesaria para otra misión. El criterio de asignación de gastos fijos se declara; el costo de oportunidad no se presenta como dinero efectivamente cobrado.

También son recursos el tiempo de espera y la atención humana. Para comparar decisiones podemos registrar varias dimensiones sin forzarlas a una única cifra monetaria. Cuando el consumo sea desconocido, se etiqueta como tal y se conserva una reserva o estimación conservadora; nunca se registra automáticamente como cero.

8.3. Selección por costo de resolución

La política busca el **costo de llegar a un resultado aceptable**, no el precio de la primera respuesta. Un intento barato que produce reintentos, errores y reconstrucción humana puede resultar peor que una intervención avanzada desde el inicio. A la inversa, una clasificación acotada no necesita consumir siempre la mayor capacidad disponible.

Tipo de trabajo	Asignación inicial propuesta
Cálculo exacto, validación o script aprobado	Software convencional cuando no haga falta interpretación
Transformación repetitiva y verificable	Ejecutor económico con desempeño comprobado en esa familia
Implementación delimitada y revisión ordinaria	Agente seleccionado por calidad y costo completo observado
Diseño experimental, contradicción difícil o fallo persistente	Capacidad avanzada desde el inicio o escalamiento justificado

No hay obligación de comenzar siempre por lo más barato. Un modelo avanzado puede descomponer un problema y delegar partes bien especificadas a ejecutores económicos. El movimiento puede ser ascendente, descendente o lateral hacia un especialista. «Mejor modelo» significa el más adecuado para el trabajo y sus restricciones, no necesariamente el más grande o costoso.

FrugalGPT y RouteLLM son antecedentes de combinación y selección de modelos para equilibrar costo y desempeño. Se citan como antecedentes de la pregunta, no como prueba de ahorro para este laboratorio. Nuestra unidad de asignación es además una misión con subtareas, revisión, recuperación y recursos compartidos. [18, 19]

La política inicial utilizará reglas explícitas y señales observables: familia de tarea, criterios verificables, pruebas fallidas, ausencia de progreso, urgencia y disponibilidad. La confianza declarada por el ejecutor no bastará para decidir que debe recibir más recursos. Todo plan incluirá reserva de verificación y una condición de parada.

8.4. Tres responsabilidades económicas

Planificación. Propone ejecutor, ruta, reserva y condiciones de escalamiento. Puede vivir en el orquestador y empezar con reglas simples; no necesita otro modelo costoso para cada llamada.

Contabilidad. Registra recursos disponibles, consumo observado y compromisos en curso. Atribuye tareas hijas, revisiones, reintentos y mejora a la misión y al proyecto correspondientes.

Control de gasto. Admite o rechaza nuevas operaciones según límites autorizados. Las reglas que se anuncien como no eludibles deben estar fuera del alcance de modificación del mismo trabajo que consume recursos.

Son responsabilidades, no tres agentes obligatorios. Su separación permite que el planificador recomiende invertir más sin poder autorizarse una ampliación. La dirección humana asigna presupuestos a proyectos y puede delegar redistribuciones dentro de bandas explícitas. El orquestador no sacrifica silenciosamente otro proyecto para terminar el actual.

8.5. Reservar, ejecutar y conciliar

Para cada presupuesto medido en una misma unidad:

Disponible = autorizado – consumo contabilizado – compromisos pendientes.

Antes de despachar trabajo se reserva su envoltorio. Al terminar se concilia con el consumo observado y se libera el remanente. La reserva se transforma en gasto, no se suma otra vez al gasto. Los cargos tardíos permanecen pendientes hasta su conciliación o tratamiento explícito.

Ejemplo hipotético: con diez unidades monetarias autorizadas, dos consumidas y seis comprometidas, quedan dos para nuevas operaciones. No quedan ocho por el hecho de que las seis todavía no aparezcan en una factura. No son precios ni saldos reales del usuario.

La admisión de compromisos concurrentes debe ser atómica: dos tareas no pueden reservar simultáneamente el mismo saldo. Cada operación tiene identidad para no duplicar cargos contables al reanudar. El presupuesto hijo es una asignación del padre, no una fuente adicional de capacidad. Una renovación de cuota puede aumentar disponibilidad según la política autorizada, pero no borra el historial de consumo.

8.6. Tope duro y presupuesto supervisado

Un **tope duro** exige poder acotar consumos en curso, considerar cargos pendientes y controlar las rutas capaces de consumirlos. Si una CLI puede realizar un número no acotado de llamadas antes de devolver el control, contar una invocación no equivale a limitar cada llamada interna.

Un **presupuesto supervisado** observa, reserva y detiene nuevas operaciones dentro de las capacidades disponibles, pero no promete impedir cualquier sobrepaso. La documentación de créditos de ChatGPT reconoce saldos negativos cuando trabajos iniciados con saldo positivo siguen consumiendo recursos concurrentemente. Consultar el saldo antes de empezar no es, por sí solo, un límite garantizado. [21]

Para cada recurso se declarará el modo realmente implementable. Ante incertidumbre se utilizarán márgenes conservadores, concurrencia y reintentos acotados, sin convertir una estimación en una garantía. El Anexo C ilustra ambos niveles; no los implementa.

Al agotarse la envoltura, el sistema puede pausar, esperar una renovación autorizada, proponer un alcance distinto o solicitar presupuesto adicional. **No puede rebajar silenciosamente los criterios de aceptación ni omitir verificaciones para declarar éxito.** La cancelación impide nuevos compromisos y conserva los que todavía podrían liquidarse.

8.7. Presupuesto para trabajar, comprobar y mejorar

Dentro de cada proyecto se separan asignaciones para producción, verificación y mejora del proceso. Pueden redistribuirse dentro de lo aprobado, manteniendo el cierre verificable de los trabajos activos. Aprender es una inversión con presupuesto propio, no una razón para consumir indefinidamente el fondo común.

El addendum advierte que Harness² consume recursos tanto en ejecuciones como en etapas de mejora y agregación de candidatos; identifica una composición desde 2k candidatos, cada uno con su ejecución. Este detalle se conserva como reporte del addendum y obliga a medir el árbol completo, no solo la tarea final. Un parámetro $k = 1$ no debe interpretarse como una única llamada. El addendum sitúa las ganancias reportadas por los autores en GLM-5 y Gemini 3.5 Flash, no en los ejecutores de nuestro laboratorio. [15, pp. 8–11]

La pregunta económica de la mejora es si el trabajo posterior justifica lo invertido, a calidad comparable. Se reportan por separado el costo de aprender, el de aplicar la política y el beneficio observado en tareas nuevas. Ahorros futuros supuestos no se contabilizan como recursos ya recuperados.

8.8. Aprender dónde invertir sin aprender a gastar sin límites

Las lecciones económicas ingresan al mismo Registro de afirmaciones: «este ejecutor basta para esta familia», «este patrón requiere escalamiento» o «esta revisión evita defectos» comienzan como hipótesis. Para volverse política necesitan evidencia, alcance, validación y autorización.

Cambios de modelo, precio, cuota, herramienta o tipo de tarea pueden invalidar una regla. Durante una evaluación formal la política queda fijada; los candidatos de mejora se estudian fuera de esa evaluación. El componente que aprende a asignar recursos puede proponer un cambio de presupuesto, pero no aumentar el presupuesto que administra.

La asignación entre proyectos debe respetar prioridades humanas, incluida investigación incierta cuyo valor no se reduce a retorno inmediato. El propósito del fondo no es convertir toda curiosidad en una cuenta de resultados. Es hacer visible qué capacidad invertimos, para qué y con qué consecuencias.

El borde económico no solo protege frente al gasto. Hace posible dedicar la mejor capacidad disponible a los problemas que realmente la merecen.

9. Autonomía útil, fronteras reales

El documento fundacional de BORDER distingue información leída de evidencia habilitada para gobernar acciones. Este white paper adopta esa distinción como principio de diseño para el laboratorio. No afirma que el gate experimental de BORDER proteja automáticamente un computador con herramientas y credenciales reales. [5]

9.1. El orquestador también debe tener una frontera

Un agente no debería poder ampliar sus permisos, aceptar su propia excepción o modificar el verificador para hacer pasar su trabajo. La implementación propuesta separaría la decisión de solicitar una operación del mecanismo que puede autorizarla.

Cuando se prometa una restricción no eludible, esta deberá apoyarse en aislamiento y permisos efectivos: credenciales fuera del entorno modificable, ejecutores con alcance delimitado y una política que el trabajo delegado no pueda reescribir. Si un agente conserva acceso irrestricto al sistema anfitrión, un wrapper o un manifiesto editable no constituyen esa garantía.

La confianza expresada por Jev o por un revisor no amplía la autoridad de una fuente. Del mismo modo, que dos modelos coincidan no demuestra independencia de sus errores. La aceptación debe conservar la relación con pruebas, entradas y efectos permitidos.

9.2. Preautorizar lo normal; escalar lo excepcional

Proponemos autorizar clases de operaciones dentro de un encargo: leer el repositorio acordado, editar una rama de trabajo, ejecutar tests definidos y producir informes. Así el humano no tiene que confirmar cada comando de una secuencia legítima.

Cambiar la pregunta experimental, ampliar acceso a datos, instalar componentes fuera del entorno autorizado, aumentar presupuesto o publicar resultados son transiciones distintas. Su tratamiento depende del contrato: pueden estar previamente autorizadas con límites o requerir una decisión explícita. No se presume autorización general por haber pedido “resuelve la tarea”.

9.3. Riesgos concretos y respuestas propuestas

Riesgo	Respuesta de diseño
Dos agentes modifican el mismo estado	Copias de trabajo separadas, bloqueos y un responsable por tarea
Una credencial vencida bloquea una corrida	Pausa identificada, reautorización y reanudación sin duplicar efectos
Se pierde el orquestador	Estado persistido y reconstrucción desde el último punto comprobado
Un archivo induce instrucciones ajenas al encargo	Tratar su contenido como datos, limitar herramientas y comprobar efectos
Una revisión confirma el sesgo del autor	Tests externos al cambio, criterios previos y disidencias registradas
La recuperación consume recursos indefinidamente	Presupuesto heredado, reservas, reintentos y condición de parada
El mejorador debilita su propia admisión	Política y verificador fuera de su superficie editable

Riesgo	Respuesta de diseño
Dos tareas reservan el mismo saldo	Libro compartido y admisión atómica de compromisos
La memoria conserva una regla obsoleta	Versiones de dependencia, suspensión y revalidación

Grok documenta que los bots de una cuenta comparten archivos y sesiones. Por ello, la separación por nombre o carpeta se tratará como organización, no como aislamiento de seguridad. [7]

La frontera no busca volver pasivo al sistema. Busca hacer posible una autonomía útil cuyo alcance pueda comprenderse, probarse y revocarse.

10. Qué existe y qué falta demostrar

La arquitectura nace de trabajo realizado, pero su declaración no debe borrar la distancia entre una experiencia parcial y un sistema integrado. Esta es la clasificación de evidencia actualizada para la versión 0.2.

Elemento	Estado de evidencia	Alcance de la afirmación
<i>Cuando nos encontramos</i>	Antecedente bibliográfico verificado	Obra de colaboración; no cotejo íntegro de sus anticipaciones [1]
Continuidad Codex CLI → Grok Bot	Reportada por el autor	Desarrollo y corridas entre herramientas; no auditoría independiente [4]
BORDER	Especificación y manuscrito aportados	Existencia de artefactos; no reejecución del estudio en este white paper [5, 14]
Computador de Grok y acceso no interactivo de Codex	Capacidades documentadas	Base técnica para intentar su conexión, no integración comprobada [6–9]
Invocación automática Grok → Codex	Propuesta	Requiere prueba de extremo a extremo y registros
Jev dentro del flujo	Acceso reportado; función propuesta	No hay beneficio experimental establecido [4, 10]
Publicación coordinada en Hugging Face	Destino propuesto	No se afirma que este flujo ya despliegue automáticamente
Controles completos del laboratorio	Diseño propuesto	Deben implementarse y probarse fuera del alcance del agente
Inspección de Harness ²	Reportada por el addendum	Commit identificado; no nueva auditoría ni corrida en v0.2 [15]
Gate de lecciones y persistencia	Propuesta integrada	No implementados ni evaluados en este documento
Fondo y frontera económica	Propuesta del autor desarrollada	Sin acceso a saldos, reservas reales ni control desplegado [16]
Mejoras de costo, utilidad o aprendizaje	Hipótesis	Requieren los contrastes separados de \$11

10.1. Qué aporta el caso BORDER

BORDER es un primer dominio apropiado porque conecta especificación, software, experimentación y análisis. El manuscrito aportado describe un benchmark diagnóstico, resultados almacenados y regeneración de tablas. Para este white paper eso constituye un antecedente de método y de artefactos; no una evaluación de la arquitectura de orquestación completa. [14]

Se evita trasladar las cifras del benchmark a una afirmación sobre productividad multiagente. Tampoco se presenta su instrumento de admisión de evidencia como una solución lista para proteger todos los efectos del laboratorio. Son objetos de evaluación diferentes.

El manuscrito está marcado como material de revisión. Esta versión no reproduce sus tablas ni resultados detallados y no autoriza su difusión. Una edición pública del white paper deberá respetar el estado editorial que el autor determine para ese trabajo.

10.2. El primer criterio de integración

La primera demostración de la arquitectura debería poder seguirse desde un encargo autorizado hasta una entrega verificada: el orquestador invoca al especialista, recibe evidencia, ejecuta una comprobación y devuelve un resultado con sus referencias. Una segunda persona debería poder reconstruir esa secuencia sin acceso al historial privado de los agentes.

Ese hito no probaría todavía una ventaja estadística, pero sí convertiría una posibilidad técnica en una instancia operativa del patrón declarado.

10.3. Qué no se transfiere por analogía

El addendum reporta software y resultados de terceros; no demuestra que sus ganancias se trasladen a los dominios del laboratorio. La admisión de lecciones no protege por sí sola un computador con credenciales reales. Una política económica propuesta no garantiza límites que el proveedor o el adaptador no permiten imponer. Estas diferencias se conservan aunque el diseño conceptual resulte convincente. [15, pp. 10–11]

11. Beneficio esperado y agenda de evaluación

La hipótesis central es que una organización explícita de capacidades puede aumentar el trabajo útil que una persona realiza sin aumentar proporcionalmente su carga de coordinación. Esto debe contrastarse: un sistema con más agentes también puede añadir demoras, errores de transferencia y costos de revisión.

Proponemos evaluar tareas representativas de tres familias: implementar y comprobar un cambio de software; ejecutar y analizar un experimento reproducible; y producir un documento fundamentado en fuentes. Los criterios de éxito se fijarán antes de observar los resultados. Un borrador convincente que no cumple esos criterios cuenta como fallo, no como éxito aproximado decidido retrospectivamente.

11.1. Comparación que separe coordinación y diversidad

	Ejecutores homogéneos	Ejecutores heterogéneos
Coordinación humana	Persona organiza trabajos con el mismo modelo disponible	Persona distribuye manualmente entre especialistas
Coordinación mediante orquestador	Orquestador utiliza el mismo conjunto homogéneo	Orquestador utiliza especialistas diferentes

Este diseño propuesto permite distinguir el efecto de automatizar la coordinación del efecto de combinar capacidades distintas. Añadiríamos una referencia simple de humano con un solo agente cuando sea pertinente. Todos los grupos deben disponer de herramientas, entradas y oportunidades de verificación comparables; no se fortalecerá artificialmente al sistema propuesto retirando capacidades esenciales a sus rivales.

11.2. Qué medir

Utilidad y calidad: tareas aceptadas, defectos relevantes, errores no detectados y fidelidad a las fuentes. **Atención humana:** minutos activos, intervenciones y trabajo de reconstrucción del contexto. **Recursos:** tiempo total, espera, cómputo, consumo reportado y costo. **Continuidad:** recuperación después de interrupciones y éxito al sustituir ejecutores. **Control:** efectos fuera del contrato y completitud de la evidencia.

Se informará el costo total y el costo marginal por separado. Una cuota incluida en una suscripción no se declarará cómputo gratuito; si una plataforma no expone su consumo, ese dato se marcará como desconocido. Los gastos fijos, la capacidad disponible y los reintentos deberán hacerse visibles.

Las repeticiones y los intervalos se definirán sobre tareas o familias independientes. No se tratarán variaciones casi idénticas como si fueran evidencia independiente ilimitada. Los umbrales, exclusiones y criterios de parada se registrarán antes de las corridas de evaluación.

11.3. Cuándo la hipótesis perdería apoyo

El beneficio quedaría cuestionado si la coordinación automática no reduce trabajo humano a calidad comparable; si la diversidad agrega costo sin mejorar resultados; si la recuperación duplica efectos; o si aceptar entregas exige reconstruir manualmente todo el proceso.

También examinaremos si el humano conserva capacidad de explicar el resultado y reconsiderar el objetivo. El programa no pretende optimizar una automatización que produzca más artefactos a costa de volver ininteligible el trabajo para quien lo dirige.

11.4. Evaluar memoria persistente y admisión

Se incorpora el diseño A/B/C del addendum sobre un **flujo de tareas**, no sobre una única respuesta. Los tres grupos deben utilizar presupuestos, conjuntos de herramientas, criterios y oportunidades de verificación comparables. [15, pp. 6–8]

Condición	Qué conserva	Pregunta que ayuda a responder
A · Reinicio	Biblioteca reiniciada en la unidad prevista por el protocolo	¿Qué ocurre sin acumulación entre barridos?
B · Persistente sin gate	Experiencias entre tareas con el control estructural descrito	¿Qué aporta o perjudica la acumulación sin admisión adicional?
C · Persistente con gate	Misma persistencia, más admisión, vigencia y promoción	¿La frontera mejora la utilidad de la memoria?

El addendum describe A como la línea base distribuida por Harness². La unidad exacta de reinicio — corrida, barrido o dominio— se comprobará contra la revisión fijada antes de ejecutar, para no convertir A inadvertidamente en un rival distinto. C no recibirá fuentes privilegiadas ni un juez mejor que B sin declararlo como otra intervención.

Se conservará el mismo flujo emparejado entre condiciones y se introducirán cambios deliberados de herramienta, modelo o interfaz a mitad del recorrido, con ramas sin deriva cuando el presupuesto lo permita. La deriva es una intervención para que envejecimiento y validez sean medibles, no una afirmación de que toda memoria persistente deba degradarse.

Predicción del addendum	Contraste previsto	Resultado que la debilitaría en el flujo estudiado
P1 · Persistencia sin gate se degrada	B frente a A a lo largo del flujo	B iguala o supera a A sin degradación relevante
P2 · El gate recupera utilidad	C frente a B, especialmente tras la deriva; también frente a A	C no mejora B o el costo elimina su beneficio
P3 · Suficiencia mecánica aporta sobre el prompt	Umbral de contrastes frente a instrucción narrativa	Descarta lecciones útiles sin reducir errores relevantes
P4 · Puede haber sobrecarga	Tareas aceptadas junto con calidad de memoria	C conserva o mejora utilidad sin bloqueo innecesario apreciable

Se mantiene P4 como un riesgo medible, no como un fracaso que deba ocultarse. La afirmación «recupera y supera la línea base» requiere comparar C tanto con B como con A; superar solo B no demuestra lo segundo. Un efecto no detectado no prueba equivalencia sin un margen y potencia acordados.

El protocolo fijará orden con semilla, número de flujos, magnitud de deriva, umbrales, exclusiones y análisis antes de mirar resultados. El addendum propone bootstrap y permutación. En v0.2 se explicita que el remuestreo debe respetar la dependencia temporal: se trabajará sobre flujos o bloques justificados, no sobre cada tarea correlacionada como si fuera una réplica independiente.

Se incorpora su revisión ciega de lecciones admitidas y rechazadas. **La calibración pertenece al piloto o desarrollo.** Si una inspección de la evaluación revela un defecto, se reporta; corregirlo genera una versión nueva y otra evaluación. No se ajustará el gate a los mismos resultados con los que luego se declara su éxito. Esta separación es una precisión metodológica de v0.2, registrada en el Anexo E.

El juez no debe recibir etiquetas de condición innecesarias ni retroalimentar al mejorador con respuestas de evaluación reservadas. Configurar modelos diferentes para ejecutar, mejorar y juzgar ayuda a separar roles, pero por sí solo no garantiza cegamiento ni independencia. Las tareas profesionales del trabajo de origen no prueban transferencia a BORDER, Seed Model o investigación propia. [15, pp. 7–11]

11.5. Evaluar la asignación económica

La hipótesis económica es que una asignación explícita y adaptable mejora el trabajo útil obtenido de un presupuesto sin degradar aceptación ni trasladar costos ocultos al humano. Se compararán una referencia con ejecutor avanzado, una política económica fija y una política adaptativa, usando las mismas tareas elegibles y restricciones.

Se evaluarán dos cortes: **calidad comparable con distinto consumo** y **presupuesto comparable con distinta utilidad**. Las comparaciones incluirán coordinación, revisión, intentos fallidos, recuperación, cómputo y mejora amortizada mediante un criterio declarado. El uso de cuotas ya contratadas no se presentará como costo total nulo.

Además de costo por tarea aceptada, se informarán gasto agregado, tareas no resueltas, defectos no detectados, intervenciones humanas, latencia, consumo desconocido y pausas por falta de recursos. Medir solo los éxitos ocultaría el costo de elegir mal. También se registrarán sobrepasos y diferencias entre reservas estimadas y liquidaciones observadas.

Las fallas deliberadas incluirán compromisos concurrentes, cargos tardíos, cuotas compartidas por interfaces distintas, credenciales vencidas, cancelación de descendientes, salidas sin consumo reportado y cambios de precio o disponibilidad. Un test sintético que impide sobregiro contable no demuestra por sí solo que el proveedor nunca facture más de lo reservado.

La política adaptativa tendrá una revisión fijada durante la evaluación. Su entrenamiento o ajuste utilizará tareas distintas; los cambios en vivo se estudiarían después bajo un protocolo propio. Primero separaremos coordinación, memoria y economía en experimentos acotados; no atribuiremos una mejora conjunta a un mecanismo que no aislamos.

11.6. Lo que puede hacer perder apoyo a v0.2

Además de los criterios de v0.1, el beneficio perdería apoyo si las lecciones persistentes deterioran tareas posteriores; si el gate protege indicadores de memoria a costa de utilidad; si enrutar consume más que lo ahorrado; si la reducción de gasto depende de omitir revisión; o si el humano debe reconstruir el trabajo para comprender qué se aceptó.

Se distinguirán cuatro logros: el circuito funciona; el aprendizaje aporta; la admisión aporta; y la asignación económica aporta. Ninguno toma prestada evidencia de los otros. La evaluación del conjunto solo se justifica después de poder inspeccionar esos mecanismos y sus costos.

12. Del primer circuito a una capacidad reutilizable

El desarrollo debe avanzar por evidencia de funcionamiento, no por cantidad de integraciones. Proponemos cuatro hitos sin fijar calendarios antes de conocer sus condiciones de ejecución.

Hito 1 • Delegación verificable. El orquestador ejecuta una tarea acotada con Codex, primero por el circuito disponible y, en la ruta propuesta por Claude, mediante Harness². Se observa el bucle completo y su experiencia antes de construir el gate de lecciones. Se utiliza biblioteca de prueba, presupuesto previo y controles básicos de acceso; «sin gate» no significa sin límites. La verificación la realiza un ejecutor distinto del productor. El Anexo D conserva el encargo concreto.

Hito 2 • Continuidad y recuperación. Interrumpir, reanudar y sustituir ejecutores sin repetir efectos ni duplicar biblioteca o asientos económicos. Se prueban cancelación, cuotas desconocidas, concurrencia y credenciales vencidas, no solo el recorrido ideal.

Hito 3 • Experimento completo. Se fija una misión, se conservan pruebas y corridas, y se regeneran tablas. El addendum propone aquí A/B/C con deriva. Los contrastes económicos se agregan como protocolos separados para no confundir causas. Jev participa solo en intervenciones que permitan medir su aporte. Las políticas de mejora no se ajustan usando la evaluación que pretende juzgarlas.

Hito 4 • Publicación controlada. Una revisión identificada llega al destino aprobado y produce evidencia de despliegue. La versión pública puede ejecutarse sin exponer credenciales personales ni conectar a visitantes con la sesión privada del laboratorio.

12.1. Implementación mínima

La primera versión puede apoyarse en una cola de tareas sencilla, un registro de estado persistente, adaptadores para herramientas y una estructura de artefactos. No requiere construir desde el inicio un framework universal, un mercado entre agentes ni un sistema de planificación aprendido. Un inventario de recursos, reservas y reglas de parada sí forma parte del circuito mínimo de v0.2.

La estabilidad de los contratos importa más que la sofisticación del coordinador. Un adaptador puede invocar una CLI; otro, una API; otro, un programa local. Cuando cambie un proveedor, deben cambiar las capacidades documentadas y la configuración, no la historia del proyecto.

El computador compartido necesita disciplina de concurrencia. Las tareas independientes pueden paralelizarse dentro de los recursos disponibles; los cambios sobre el mismo repositorio o entorno deben coordinarse. Las cuotas y los límites efectivos se medirán antes de aumentar la carga.

12.2. Un laboratorio que beneficia a otros proyectos

BORDER es el primer caso, no la frontera de aplicación. Seed Model podría reutilizar contratos de corrida, registros y reproducción sin modificar por ello las hipótesis de su simulador. El trabajo editorial podría conservar fuentes, objeciones y decisiones de voz mientras automatiza tareas mecánicas. Un prototipo de producto podría pasar de una necesidad a un artefacto probado bajo el mismo patrón de delegación.

Estas son aplicaciones propuestas. Su beneficio tendría que demostrarse en cada dominio. Lo reutilizable es la organización del trabajo; los criterios de validez siguen siendo propios de cada proyecto.

La ambición no es que el laboratorio haga todo. Es que cada aprendizaje útil pueda incorporarse sin empezar de nuevo, y que cada nuevo componente amplíe capacidades sin borrar el criterio que las orienta.

12.3. Un primer paso proporcionado

La propuesta de Claude reduce una incógnita: observar una experiencia real antes de especificar todas sus reglas de admisión. El primer hito no exige resolver por adelantado la optimización del fondo ni la generalización de lecciones. Sí exige declarar qué se permitirá ejecutar y cuánto se puede comprometer. Después se decide qué mecanismo merece el siguiente presupuesto.

13. Declaración de arquitectura

Declaramos como dirección de trabajo una inteligencia simbiótica distribuida: una colaboración en la que la persona no queda reducida a operadora de interfaces ni los agentes a productores de respuestas aisladas. Su propósito es sostener ciclos completos de investigación, creación y construcción.

La voluntad humana conserva la facultad de preguntar, valorar, corregir y cambiar de rumbo. Decide qué resultado importa, qué explicación es suficiente y qué proyecto merece continuar. Puede aprender de las objeciones y resultados de los agentes sin transferirles la autoridad sobre sus fines.

El orquestador transforma un encargo en trabajo coordinado. Recurre a agentes y programas, solicita correcciones y gestiona recuperación dentro del alcance acordado. No necesita resolverlo todo internamente ni puede fabricar la evidencia de su éxito.

La memoria permite que el trabajo sobreviva a las conversaciones. Las decisiones conservan razones y fuentes; los resultados, sus condiciones de producción; las discrepancias, su lugar en el registro. El aprendizaje modifica procedimientos cuando la evidencia y la autoridad lo permiten, no cuando una explicación parece convincente.

La arquitectura es abierta: Grok, Codex, Claude, Astra, Jev, los programas convencionales, GitHub y Hugging Face son una configuración sustituible. Ningún proveedor define por sí solo al laboratorio. Incorporar una capacidad debe ampliar lo que podemos hacer sin borrar la continuidad del proyecto.

La economía convierte recursos disponibles en decisiones de inversión. Resolver, verificar y mejorar comparten un fondo finito. El sistema puede aprender a utilizarlo mejor; no apropiarse del derecho a ampliarlo ni ocultar sus costos.

La autonomía se evalúa por el trabajo legítimo completado, no por actuar sin humanos a cualquier precio. Un sistema que nunca actúa tampoco satisface esta propuesta. La evidencia limita las afirmaciones: trazabilidad, reproducción, comparación e integración son logros distintos, no intercambiables.

Del libro a una práctica que puede ponerse a prueba

Cuando nos encontramos abrió una exploración de la creación entre inteligencias. Esta arquitectura la prolonga: conversar para formular, delegar para construir, ejecutar para observar y volver al diálogo para comprender. No declara cumplidas todas sus anticipaciones; organiza una práctica para descubrir hasta dónde puede llegar esa orientación. [1]

No buscamos una inteligencia que lo haga todo por nosotros. Buscamos una organización de inteligencias con la que podamos hacer cosas que antes no estaban a nuestro alcance.

Así definimos el simbiote que queremos construir: una voluntad humana que reúne las capacidades adecuadas, organiza su cooperación, aprende con ellas y conserva el derecho de cambiar el rumbo. Su continuidad no depende de que un modelo dure para siempre, sino de que el propósito, la memoria y lo comprobado puedan seguir creciendo juntos.

Anexo A. Contrato de misión y devolución

Esquema ilustrativo, normalizado para discusión. No es una implementación desplegada. Las referencias no resueltas impiden pasar de borrador a ejecución autorizada.

```

schema_version: simbionte-task/0.2
mission_id: identificador_estable
task_id: identificador_de_tarea
parent_task_id: null
objective: resultado_y_alcance_aprobados
inputs:
  source_refs: [referencias_autorizadas]
  code_revision: revision_fijada
  protocol_revision: protocolo_aplicable
  operational_library_revision: biblioteca_admitida
capabilities: [subconjunto_autorizado]
policy_ref: permisos_y_criterios_protegidos
budget_ref: asignacion_del_presupuesto_padre
routing_policy_revision: politica_de_asignacion_fijada
acceptance:
  criteria_refs: [criterios_versionados]
  verifier_role: distinto_del_productor
artifacts: [productos_y_registros_requeridos]
recovery:
  retry_limit: definido_antes_de_correr
  cancellation: propagar_a_descendientes
learning:
  candidate_output: permitido
  persistent_promotion: autorizacion_separada
publication: autorizacion_separada

```

La devolución identifica tarea e intento, estado, revisiones, artefactos, comandos y comprobaciones, errores no resueltos, consumo observado o desconocido y reservas por conciliar. Una lección candidata apunta a su evidencia, no cambia por sí misma la biblioteca operativa. El mensaje no requiere revelar razonamientos internos privados de los modelos.

Aceptación mínima. Fidelidad al encargo; objetos inspeccionables; comprobación sobre la revisión entregada; permisos y presupuesto heredados; reanudación sin efectos duplicados; y publicación solo bajo autorización. El analizador de resultados es un proceso distinto del mensaje narrativo del ejecutor.

Anexo B. Registro de lecciones y promoción

Se conservan las claves cue, lesson y evidence del addendum, ampliadas con linaje, condiciones y autorización. Este formato propuesto no afirma compatibilidad directa con la biblioteca de Harness². El adaptador deberá verificarla.

```

schema_version: simbiote-lesson/0.2
lesson_id: identificador_estable
cue: condicion_observable_de_aplicacion
lesson: cambio_propuesto_y_alcance
evidence:
  supporting_refs: [registros_y_contrastes]
  counterevidence_refs: []
  independent_task_refs: [tareas_de_confirmacion]
binding:
  component: herramienta_prompt_o_politica
  before_revision: revision_previa
  after_revision: revision_candidata
validity:
  produced_at: instante_de_la_evidencia
  domain: familia_de_tareas
  dependencies: [modelo_herramienta_protocolo]
  invalidate_on: [cambios_relevantes]
claims_register_state: reportada
operational_status: candidata
admission:
  policy_revision: regla_de_admision
  verifier_ref: comprobacion_independiente
  decision: pendiente
promotion:
  authorized_by: null
  library_revision: null
supersedes: null

```

Transición mínima. Proponer → registrar → comprobar → reproducir → autorizar → activar. Una comparación posterior estima el beneficio, sin extender automáticamente el dominio de validez. Caducar, suspender, revocar o sustituir son transiciones operativas con razones conservadas.

Pruebas indispensables. Una lección sin registros no se promociona; leerla hoy no rejuvenece su evidencia; cambiar una dependencia relevante activa revalidación; reglas contradictorias se exponen; la promoción no puede modificar su propio verificador; el rollback conserva historia; y la escasez presupuestaria no reduce el estándar de admisión.

Anexo C. Contrato económico y libro de recursos

El ejemplo utiliza importes hipotéticos. No son tarifas, saldos ni una autorización de gasto. supervised expresa que todavía no se ha demostrado un tope duro para el recurso externo.

```

schema_version: simbionte-budget/0.2
budget_id: presupuesto_mision_ejemplo
parent_budget_id: fondo_proyecto_ejemplo
authority_ref: aprobacion_humana_pendiente
currency: USD
cash_limit: 10.00
allocations:
  production: 6.00
  verification: 2.00
  improvement: 2.00
resource_accounts:
  - quota_id: cuota_compartida_A
    access_mode: authorized_subscription
    native_unit: provider_quota
    observed_remaining: null
    usage_status: unknown
    renewal_policy_ref: condiciones_a_verificar
controls:
  enforcement_mode: supervised
  include_descendant_spend: true
  atomic_reservations_required: true
  unknown_usage_is_zero: false
  new_budget_requires_authorization: true
  protected_verification_reserve: true
on_exhaustion:
  - pause_new_commitments
  - reconcile_pending_usage
  - request_scope_or_budget_decision

```

Cada asiento registra event_id, mission_id, tarea e intento, recurso, unidad nativa, importe reservado, consumo liquidado, estado, tiempo y fuente de medición. El libro permite reservar, liquidar, liberar, corregir y conciliar. Las correcciones no eliminan el asiento anterior.

Para la moneda del ejemplo: autorizado 10; consumido 2; comprometido 6; disponible 2. Las cuotas nativas se verifican adicionalmente en sus propias unidades. Suficiente dinero no implica suficiente cuota; una cuota renovada no implica nuevo permiso de gasto monetario.

La autoridad aprueba límites y posibles redistribuciones. El planificador selecciona rutas dentro de ellos. Un aviso de presupuesto no se denomina bloqueo efectivo hasta probar que intercepta todas las rutas relevantes, incluidos trabajos internos y concurrentes.

Anexo D. Primer encargo: observar antes de construir el gate

Contrato derivado del addendum, pp. 8–9, con sintaxis normalizada y referencias económicas añadidas. Mantiene su prioridad: correr primero el mecanismo candidato en un entorno de prueba, sin gate de lecciones ni promoción hacia la biblioteca productiva. No se ejecuta por estar incluido en este documento.

```
mission_id: simbionte-harness-admision
task_id: h1-delegacion-verificable
parent_task_id: null
status: draft_requires_authorization
objective: >
  Ejecutar una tarea acotada de JobBench con Harness2 y
  su ejecutor Codex. Devolver harness, experiencia y logs.
inputs:
  source_ref: google-research/harness2
  code_revision: 1cfe254e8d25b15f37048c3730134686abe60335
  protocol_revision: h1-v02-draft
roles:
  orchestrator: Grok_Bot
  engineering_executor: Codex_CLI
  artifact_verifier: Claude_Code_o_equivalente_distinto
preflight:
  - comprobar_revision_interfaz_y_credenciales_admitidas
  - declarar_presupuesto_y_consumos_internos_posibles
  - revisar_superficie_editable_y_permisos
  - ejecutar_dry_run_y_conservar_seleccion
scope:
  benchmark: jb
  k: 1
  runs_dir: directorio_exclusivo_de_prueba
  operational_library_write: false
  admission_gate: no_en_esta_tarea
budget_ref: presupuesto_h1_pendiente_de_aprobacion
forbidden:
  - ampliar_permisos_o_presupuesto
  - modificar_controles_protegidos
  - publicar_o_exponer_credenciales
retry_limit: 2
publication: autorizacion_separada
```

El comando de previsualización consignado en el addendum es `harness2 --bench jb --k 1 --dry-run`. Se verifica contra la revisión fijada antes de usarlo. La aceptación requiere selección conservada sin llamadas durante esa previsualización, registros de la ejecución real, árbol del harness adoptado y experiencia con sus campos. Un segundo proceso reconstruye el resumen; un lector externo puede seguir la secuencia sin el chat privado. [15, pp. 8–9]

Si no se produce una experiencia, se conserva el diagnóstico y se marca el criterio como incumplido; no se fabrica el archivo para satisfacer el contrato. Ante acceso vencido, se pausa para reautorización. Ante fallo, se respetan los dos reintentos acumulados dentro del mismo presupuesto. El uso de credencial API o suscripción se comprueba en el adaptador: soporte de Codex no demuestra automáticamente equivalencia entre ambos accesos.

Anexo E. Cambios y preguntas para la revisión con Claude

E.1. Qué cambia respecto de v0.1

Elemento	Tratamiento en v0.2
Centro del sistema	Voluntad humana, propósito revisable y mejores capacidades para cada trabajo
Cinco planos y memoria	Se conservan; aprendizaje y economía se integran como responsabilidades transversales
Aprendizaje del proceso	Se incorpora el addendum y Harness ² como candidato; no se declara integrado
Registro de afirmaciones	Se aplica a lecciones y políticas económicas, con promoción autorizada
Herramientas y revisión	Se incorpora Claude Code por rol; se explicita el conflicto de revisar propuestas propias
Recursos	Fondo heterogéneo, costo total/marginal/oportunidad y presupuestos heredados
Control económico	Reserva, ejecución, conciliación, concurrencia y distinción entre tope y supervisión
Evaluación	Coordinación/diversidad, A/B/C con deriva y comparación de políticas económicas
Hitos y contratos	Primer circuito sin gate de lecciones, con controles básicos y presupuesto previo

E.2. Precisiones editoriales explícitas

Se preserva el corte del addendum entre observada y reproducida, pero se distingue elegibilidad de autorización. Se conservan P1–P4, acotando sus conclusiones al flujo estudiado. La revisión ciega calibra en desarrollo; no habilita corregir sobre la evaluación y reportarla como intacta. La configuración de varios jueces no equivale por sí sola a independencia o cegamiento. Estas son precisiones de v0.2 para discusión, no conclusiones atribuidas retrospectivamente a Claude.

Las referencias a la inspección de Harness² conservan la atribución al addendum y su commit. Esta versión no ha vuelto a auditar ese código, ejecutado sus pruebas ni confirmado acceso por suscripción dentro de su adaptador. El texto del libro tampoco se revisó íntegramente: no se añaden predicciones específicas no cotejadas.

E.3. Preguntas de revisión prioritaria

R1 • Identidad y voluntad. ¿La definición conserva la colaboración y la posibilidad de que el humano aprenda y reconsidere, sin reducirlo a operador ni convertir la metáfora en una afirmación de conciencia colectiva?

R2 • Admisión. ¿El corte hacia política persistente es suficiente para el piloto? ¿Qué número de contrastes y qué independencia exige cada tipo de lección? ¿Qué parte puede comprobar software y qué parte necesita juicio?

R3 • Superficie protegida. ¿Qué archivos y capacidades concretos no puede modificar el bucle? ¿El adaptador conserva esa separación, incluida la posibilidad de que una herramienta evada un wrapper?

R4 • Integración y acceso. ¿El substrate fijado puede usar el mecanismo autorizado disponible? ¿Qué consumo interno genera realmente $k = 1$? ¿Qué requiere un adaptador de Claude Code contra la interfaz descrita?

R5 • Economía. ¿Cómo se identifican cuotas compartidas y compromisos concurrentes? ¿Qué recursos permiten un tope duro y cuáles solo supervisión? ¿Cómo se reserva la verificación cuando falta telemetría?

R6 • Evaluación. ¿A/B/C permite aislar persistencia, admisión y sobrebloqueo? ¿Cómo se emparejan flujos, deriva y presupuestos sin contaminación? ¿Qué resultados contradirían la promesa económica?

R7 • Revisión independiente. Claude participó en el addendum y Astra en esta integración. Pueden comprobar artefactos y discutir cambios, pero ninguno debe declararse revisor adversarial independiente de su propio diseño. ¿Quién examina esas decisiones sin haberlas propuesto?

R8 • Proporción y primer paso. ¿El Hito 1 permite aprender algo concreto sin construir prematuramente toda la arquitectura? ¿Conservamos ambición y capacidad de actuar, en vez de volver el laboratorio una colección de gates que nunca trabaja?

La siguiente iteración debería devolver objeciones ancladas a secciones, cambios propuestos y su efecto sobre contratos, evidencia y costo. No basta una valoración general favorable. La aprobación de esta versión y cualquier decisión de implementación siguen correspondiendo al autor.

Fuentes y trazabilidad

Las referencias de v0.1 conservan su procedencia. El **19 de octubre de 2026** se consultó nuevamente documentación primaria sobre Grok, Codex, Jev, GitHub Actions, Spaces y orquestación, y se incorporaron antecedentes de enrutamiento y créditos. No se auditó el código de Harness²: las observaciones sobre su commit se atribuyen al addendum. Las capacidades y condiciones de productos pueden cambiar.

Los contratos, métricas, hitos, principios, biblioteca operativa y control económico son propuestas de este white paper, salvo indicación expresa. Las referencias [15] y [16] identifican los aportes incorporados; el Anexo E explicita los refinamientos de esta edición.

Antecedentes y materiales del autor

[1] Niven, K.; Vera, P. M. (2025). *Cuando nos encontramos: Una colaboración entre humano e inteligencia artificial*. Amazon KDP. ISBN 9798289810854. Se consultaron la ficha bibliográfica y la descripción editorial en Google Books, no el texto íntegro. La cubierta disponible en los materiales del autor coincide en título y firmas. [Fuente pública](#)

[2] Simbionte. *About*. Presentación pública del proyecto como colaboración humano-sintética orientada a preguntas, reflexión y co-creación. Fuente primaria del planteamiento editorial. [Fuente pública](#)

[3] Simbionte (24 de agosto de 2026). *All the Parts Work. The System Doesn't*. Ensayo del autor sobre la diferencia entre capacidades individuales y organización del sistema. Se utiliza su argumento editorial, no se reevalúan aquí los trabajos científicos que comenta. [Fuente pública](#)

[4] Vera, P. M. Comunicaciones de trabajo en ChatGPT: conversación actual de HF Spaces/BORDER, 19 de octubre de 2026; antecedente conversacional del Directorio de IAs, 30 de enero de 2026, recuperado del historial. Fuente del reporte Codex CLI→Grok Bot y de la intención de automatizar la coordinación. No constituye auditoría de ejecución.

[5] *BORDER Protocol v0.1*. Documento fundacional aportado por el autor: propósito, precedencia de la frontera, separación de estados y criterios de automatización. Material de trabajo del proyecto.

[14] *Stale, Scoped, or Spoofed? A Diagnostic Benchmark for Evidence Admission at the Agent Action Boundary*. Archivo aportado: *BorderProtocol_V01.pdf*, siete páginas, manuscrito anónimo de revisión. Referencia interna; no se reproducen tablas ni se autoriza difusión. No se auditó el repositorio ni se repitieron corridas al elaborar este white paper.

Documentación técnica primaria

[6] xAI. *Grok Bot — Overview*. Computador persistente y operación mediante terminal, navegador y archivos. [Fuente pública](#)

[7] xAI. *Use the computer and apps*. Entorno compartido, archivos y sesiones de los bots de una cuenta. [Fuente pública](#)

[8] OpenAI. *Authentication*. Acceso por ChatGPT o API; autenticación en entornos remotos y tratamiento de credenciales. [Fuente pública](#)

[9] OpenAI. *Non-interactive mode*. Invocación mediante `codex exec`, eventos JSONL y salidas estructuradas. [Fuente pública](#)

[10] TypeScript. *Introduction*. Jev, preguntas tipadas y respuestas estructuradas consumibles por software. [Fuente pública](#)

[11] GitHub. *Understanding GitHub Actions*. Flujos de construcción, pruebas, trabajos y despliegue. [Fuente pública](#)

[12] Hugging Face. *Spaces Overview*. Demostraciones, repositorios de Spaces, ciclo de despliegue y secretos. Las condiciones de acceso y recursos se verifican al implementar; no se presupone costo nulo. [Fuente pública](#)

[13] OpenAI. *Agent orchestration — Agents SDK*. Patrones existentes de agentes como herramientas, transferencias y orquestación mediante código o modelos. Antecedente técnico; no es una dependencia obligatoria del diseño. [Fuente pública](#)

Aportes de v0.2 y antecedentes económicos

[15] Vera, P. M., con Claude (19 de octubre de 2026). *El bucle que aprende necesita una frontera*. Archivo Addendum *CLaude.pdf*, 12 páginas. Base de la integración de Harness², la promoción de lecciones, los roles sustituibles, A/B/C, el Hito 1 y sus riesgos. Se conserva la distinción entre inspección reportada de código y ejecución no realizada.

[16] Vera, P. M. (19 de octubre de 2026). Conversación de revisión de HF Spaces/Simbionte: propuesta del «fondo de tokens», criterio económico por dificultad y formulación del simbionte como voluntad que reúne y orquesta capacidades. La respuesta discutida desarrolla la asignación y la frontera; el autor solicita integrarlas en v0.2. Comunicación de trabajo, no evaluación experimental.

[17] Harness². Repositorio google - research/harness2 y Harness²: *Recursive Agent Harnessing for an Open World*, arXiv:2610.14224, según la referencia del addendum. Revisión consignada: 1cfe254e8d25b15f37048c3730134686abe60335. Licencia, adaptadores, resultados y contenido del código no se reauditaron al preparar v0.2. [Repositorio citado](#) · [Trabajo citado](#)

[18] Chen, L.; Zaharia, M.; Zou, J. *FrugalGPT: How to Use Large Language Models While Reducing Cost and Improving Performance*. arXiv:2305.05176. Antecedente de combinaciones en cascada; no evidencia del ahorro de Simbionte. [Fuente primaria](#)

[19] Ong, I. y colaboradores. *RouteLLM: Learning to Route LLMs with Preference Data*. arXiv:2406.18665. Antecedente de selección entre modelos con datos de preferencias. [Fuente primaria](#)

[20] OpenAI. *Pricing*, documentación de Codex. Diferencia tokens, créditos y consumo incluido; fuente del ejemplo de unidades heterogéneas, no de una tarifa fijada en este white paper. [Fuente primaria](#)

[21] OpenAI. *Using Credits for Flexible Usage in ChatGPT (Personal plans)*. Créditos compartidos entre funciones elegibles y posibilidad de saldo negativo por trabajo concurrente. Condiciones sujetas al plan y cuenta; créditos de producto y API no se presuponen intercambiables. [Fuente primaria](#)

El addendum cita además *The Granularity Mismatch in Agent Security* (arXiv:2605.11039) y *Proof-or-Stop* (arXiv:2607.14890). Se conservan como referencias de su discusión, sin reevaluar esos trabajos ni importar sus resultados a la arquitectura aquí declarada. [15, p. 11]

Alcance de esta versión. Se ha redactado e integrado documentación. No se afirma haber conectado cuentas, instalado ejecutores, iniciado corridas, implementado gates, administrado saldos reales, modificado repositorios o desplegado el laboratorio. La revisión del autor y la siguiente iteración con Claude permanecen abiertas.