GUÍA GRATUITA

COSAS AGÉNTICAS

Los 8 niveles de *Hermes*

Del chatbot al cerebro de empresa: el marco Contexto → Artefacto → Habilidades, desglosado nivel a nivel con comandos reales.

Marco inspirado en el vídeo de Eric Siu (Leveling Up with Eric Siu, julio de 2026) — adaptación y desarrollo propios en español. Sin afiliación con Eric Siu, Leveling Up ni Nous Research.

Edición de regalo

Sin email · Sin trampa

Agosto 2026 · cosasagenticas.com

CONTENIDO

Los 8 niveles de Hermes

De preguntar a cerebro de empresa

El marco Contexto → Artefacto → Habilidades aplicado a Hermes Agent, desglosado nivel a nivel

> *Marco inspirado en el vídeo de Eric Siu (Leveling Up with Eric Siu, julio de 2026). Adaptación y desarrollo propios en español. Los 8 niveles se mencionan en el vídeo como síntesis final; el desglose, los ejemplos y los comandos de este documento son desarrollo original de esta guía.*

Nota de no afiliación: este documento no está afiliado ni patrocinado por Eric Siu, Leveling Up, Single Grain ni Nous Research. Hermes Agent es un producto de Nous Research.

El error de usar Hermes como un ChatGPT mejorado

La mayoría de la gente que instala Hermes hace exactamente lo mismo el primer día: abre una sesión, escribe una pregunta, lee la respuesta, cierra la terminal. Al día siguiente repite el proceso desde cero. Es cómodo, funciona, y es la manera más rápida de desperdiciar la herramienta. Eric Siu lo resume sin rodeos: si usas Hermes como un chatbot estándar, probablemente estés aprovechando entre el 5 y el 10 % de su capacidad.

La diferencia no está en el modelo. Está en qué haces con lo que sale. Un chatbot produce respuestas: texto que lees, quizá copias a un documento, y que muere en el historial de una conversación que nadie más verá. Un sistema produce activos: archivos que tu equipo abre, procesos que se ejecutan solos a las nueve de la mañana, decisiones que quedan registradas y que el propio agente puede consultar seis semanas después. La primera categoría se consume. La segunda se acumula.

Piensa en un caso corriente. Una agencia pequeña dedica los lunes por la mañana a revisar el rendimiento de las campañas de sus doce clientes. Antes: alguien abre Hermes, pega los datos de un cliente, pide un análisis, lee la respuesta, la resume en un correo, y repite once veces. Tres horas y media de trabajo que se evapora en cuanto se envía el último correo. Después: existe una skill llamada `revision-campanas` que sabe qué métricas mirar y en qué formato entregarlas, un cron que la dispara los lunes a las 8:00 con los datos ya conectados, y un informe HTML por cliente que aterriza en el canal de Slack del equipo antes de que nadie encienda el portátil. El lunes siguiente el coste marginal es cero. El trabajo no desaparece: se convierte en infraestructura.

Ese salto —de consumir respuestas a acumular sistemas— tiene ocho escalones. La mayoría de la gente está en el primero.

El marco: Contexto → Artefacto → Habilidades

Toda la lógica de Hermes se sostiene sobre tres piezas encadenadas. Siu lo formula así: el modelo es contexto; lo que tengas de contexto se convertirá en un artefacto; y cuando esas acciones se vuelven repetibles, se convierten en habilidades.

Contexto es lo que el agente sabe cuando empieza a trabajar: los archivos que le has dado, las fuentes conectadas, las instrucciones que has escrito, la memoria de sesiones anteriores. Un contexto pobre produce resultados genéricos por muy bueno que sea el modelo. Un contexto rico —tus números reales, tu vocabulario, tus restricciones— produce resultados que solo tú podrías haber obtenido.

Artefacto es lo que queda cuando el agente termina: un PDF, un HTML, una hoja de cálculo, un checklist. Algo con nombre de archivo, que se abre, se comparte y se digiere rápido. La clave es que un artefacto lo consumen dos audiencias distintas: tu equipo y tus propios agentes. Un informe bien estructurado es a la vez un documento para humanos y contexto de entrada para la siguiente tarea automatizada.

Habilidad es lo que ocurre cuando has producido el mismo tipo de artefacto tres veces y decides congelar el procedimiento. Dejas de reexplicar el proceso cada vez y lo escribes una vez, en un archivo, con nombre. A partir de ahí el proceso es invocable, mejorable y —esto es lo importante— transferible: otra persona del equipo lo usa sin saber cómo lo construiste, y otros agentes pueden apoyarse en él.

Un ejemplo de marketing. Contexto: los últimos seis meses de datos de tu blog, la lista de palabras clave por las que quieres posicionar, y las tres piezas que mejor convirtieron. Artefacto: un informe HTML con las diez oportunidades de contenido ordenadas por tráfico potencial frente a esfuerzo, con el ángulo editorial sugerido para cada una. Habilidad: `auditoria-contenido`, que toma cualquier dominio y devuelve ese mismo informe con la misma estructura. Lo que la primera vez costó cuarenta minutos de ida y vuelta, la quinta vez cuesta escribir una línea.

Un ejemplo de operaciones. Contexto: el histórico de incidencias de soporte del último trimestre, el SLA que has prometido y la plantilla actual del equipo. Artefacto: un PDF de dos páginas con los cinco motivos de contacto más frecuentes, el tiempo medio de resolución de cada uno y qué tres respuestas tipo eliminarían el 40 % del volumen. Habilidad: `informe-soporte`, que se ejecuta cada mes con los mismos criterios, de modo que los informes son comparables entre sí en lugar de ser cinco documentos con cinco formatos distintos.

La portabilidad es la propiedad que hace que esto escale. Una habilidad no es una nota personal: es algo que quieres poder entregarle a otra persona. En Hermes las skills viven en `~/.hermes/skills/<nombre>/SKILL.md`, un archivo de texto con un frontmatter mínimo — `name:` y `description:` — y el procedimiento escrito debajo. Al ser un archivo, se versiona, se comparte, se copia a la máquina de otro y funciona igual.

Respuesta privada frente a artefacto para el equipo

Hay una frase del vídeo que conviene tener pegada al monitor: «Si Hermes te da una respuesta privada, eso es útil. Ahora, si le da a tu equipo un artefacto, entonces estamos hablando de leverage».

La diferencia parece cosmética y no lo es. Una respuesta privada tiene un lector, una vida útil de minutos y cero capacidad de composición: nadie puede construir encima. Un artefacto tiene un archivo, una ubicación, un formato estable y una audiencia. Se reenvía. Se comenta. Se convierte en la entrada de la reunión del martes. Y, si mañana quieres automatizar el proceso, ya tienes el formato de salida definido, que es la mitad del trabajo.

Los artefactos están, como dice Siu, muy infravalorados, y creo que la razón es psicológica: generar un archivo parece más ceremonioso que leer una respuesta en pantalla. Pero el coste real de producirlo es prácticamente nulo y el retorno se multiplica por el número de personas que lo leen. Un análisis de precios que solo tú has visto no cambia nada. El mismo análisis en un PDF de tres páginas que circula por el equipo comercial cambia cómo se cotiza la semana siguiente.

Regla práctica: si lo que estás pidiendo tiene valor para alguien más que tú, o para tu yo de dentro de un mes, pide el artefacto explícitamente. No cuesta nada añadirlo a la instrucción.

	Respuesta privada	Artefacto compartible
Lectores	1	El equipo entero
Vida útil	La sesión	Indefinida
Reutilizable por agentes	No	Sí
Base para automatizar	No	Sí
Coste de producirlo	Bajo	Prácticamente el mismo

Nivel 1 — Preguntar

Es el punto de partida de la mayoría y no tiene nada de malo, siempre que no te quedes ahí. Preguntar significa usar Hermes como un buscador conversacional: dudas puntuales, explicaciones, un borrador rápido de correo, una segunda opinión sobre una idea. La sesión empieza vacía, produce texto y termina sin dejar rastro utilizable.

En la práctica son dos comandos. Una sesión interactiva para conversar, y una respuesta única cuando solo quieres un dato sin abrir nada:

```
hermes chat
hermes -z "resume en cinco puntos las diferencias entre facturación por suscripción y por uso"
```

Si acabas de instalar Hermes, elige tu plataforma:

MACOS / LINUX · TERMINAL

```
curl -fsSL https://hermes-agent.nousresearch.com/install.sh | bash
```

WINDOWS · POWERSHELL

```
iex (irm https://hermes-agent.nousresearch.com/install.ps1)
```

¿Prefieres una app con ventana en vez de la terminal? Descarga la app de escritorio de Hermes para tu sistema.

Cuando termines, dedica diez minutos a dejar la base bien puesta antes de seguir subiendo:

```
hermes setup
hermes model
hermes doctor
```

Resultado observable: respondes preguntas más rápido que buscando en Google. Nada más. Tu operación funciona exactamente igual el mes que viene que el mes pasado. El indicador de que estás atascado en este nivel es simple: si borrases tu historial de sesiones, no perderías nada de valor.

Nivel 2 — Delegar tareas puntuales

El segundo nivel es el primero en el que hay trabajo real de por medio. En lugar de preguntar, encargas: analiza este CSV de ventas y dime qué producto está cayendo, redacta la descripción de este puesto a partir de estas notas, revisa este contrato y márcame las cláusulas de penalización. Hay un input concreto, un entregable concreto y un criterio de aceptación.

La diferencia con preguntar está en cómo formulas el encargo. Un buen encargo incluye tres cosas: el material de partida, el formato de salida y la restricción. «Analiza esto» produce un muro de texto genérico. «A partir de este CSV, dame las tres líneas de producto con mayor caída interanual, con el porcentaje y una hipótesis de causa por cada una, en menos de 200 palabras» produce algo que puedes usar.

```
hermes -z "lee ~/datos/ventas-q2.csv y dame las 3 líneas con mayor caída interanual, con
porcentaje e hipótesis de causa, máximo 200 palabras"
hermes tools list
hermes sessions list
```

Resultado observable: tareas de treinta o cuarenta minutos bajan a cinco. Un caso típico: una consultora que dedicaba dos horas a preparar el resumen previo a cada reunión con cliente pasó a veinte minutos, con ocho reuniones al mes eso son unas trece horas recuperadas. Real, pero con un techo bajo: cada tarea sigue siendo manual, y el ahorro se repite pero no crece. El trabajo sigue evaporándose en cuanto cierras la sesión.

Nivel 3 — Crear artefactos compartibles

Aquí empieza el leverage de verdad. El nivel 3 consiste en cambiar una sola cosa en tus encargos: exigir que el resultado sea un archivo con formato, no un bloque de texto en la terminal. Un PDF que se envía por correo, un HTML que se abre en el navegador y se lee en el móvil, un checklist que alguien puede tachar, una tabla comparativa que se pega en una presentación.

El motivo por el que esto multiplica el valor es que cambia el número de personas afectadas por el mismo minuto de trabajo. Ese es literalmente el significado de leverage: mismo esfuerzo, más output. Un análisis que solo tú lees ha costado lo mismo que uno que leen ocho personas, pero el segundo mueve la operación.

Hay una segunda razón, menos obvia y más importante a medio plazo: los artefactos son el formato en el que tus agentes se pasan trabajo entre sí. Un informe estructurado en HTML no es solo un documento para humanos, es contexto de entrada limpio para la siguiente tarea. Cuando llegues al nivel 5, tus crons van a producir y consumir artefactos constantemente. Si no tienes el hábito de generar archivos, no tendrás nada que encadenar.

```
hermes chat
```

Y dentro de la sesión, la instrucción concreta: «con los datos de `~/datos/ventas-q2.csv`, genera un informe HTML de una página con tres bloques —resumen ejecutivo, tabla de líneas de producto ordenadas por variación, y tres acciones recomendadas— y guárdalo en `~/informes/ventas-q2.html`».

Para las tareas de una sola pasada donde ya sabes exactamente qué quieres:

```
hermes -z "genera un checklist en markdown para el onboarding de un cliente nuevo de
consultoría, 12 puntos máximo, agrupados en tres fases, y guárdalo en ~/plantillas/
onboarding-cliente.md"
hermes sessions export
```

Resultado observable: el trabajo deja de morir contigo. Empiezas a tener una carpeta de entregables que crece, y otras personas del equipo dejan de pedirte cosas que ya están en un archivo.

Ejemplo práctico: una empresa de formación online generaba manualmente la ficha de cada curso —temario, duración, requisitos, resultados de aprendizaje— en un documento distinto cada vez. Catorce cursos, catorce formatos, y ninguna posibilidad de compararlos. Al definir un artefacto único —una ficha HTML de una página con la misma estructura— rehicieron las catorce en una tarde. El efecto secundario fue el interesante: con las fichas homogéneas, detectaron que tres cursos prometían el mismo resultado de aprendizaje con nombres distintos, y los fusionaron. El artefacto no solo comunicó mejor: reveló un problema que el formato libre escondía.

Nivel 4 — Convertir lo repetible en skills

La regla es de las más útiles del marco y no admite matices: lo que repitas más de una vez, probablemente deberías convertirlo en una skill. Segunda vez que escribes las mismas instrucciones, ya estás pagando un impuesto. Tercera vez, el impuesto es voluntario.

Una skill es un procedimiento escrito una vez y guardado en `~/.hermes/skills/<nombre>/SKILL.md`. Frontmatter con `name:` y `description:`, y debajo el proceso: qué entrada espera, qué pasos sigue, qué formato tiene la salida, qué comprobaciones hace antes de darse por terminada. No es un prompt largo: es documentación de un proceso, escrita para que la ejecute un agente.

Los prompts son desechables; las skills, no. Y hay una propiedad que las hace especialmente rentables: **componen**. Una skill de investigación de competencia puede alimentar a una de posicionamiento, que a su vez alimenta a una de generación de mensajes de venta. Cada una es simple y verificable por separado, pero encadenadas producen algo que sería inmanejable como prompt único. Es la misma lógica que las funciones en programación: la ganancia no está en la primera, está en la décima, cuando ya puedes combinarlas.

Antes de escribir una skill desde cero, mira si ya existe. El ecosistema tiene registros públicos y no tiene sentido reconstruir lo que alguien ya resolvió:

```
hermes skills search auditoria
hermes skills inspect auditoria-seo
hermes skills install auditoria-seo
hermes skills list
```

`inspect` es el paso que la mayoría se salta y no debería: te permite ver qué hace una skill antes de instalarla. Instalar código de terceros sin leerlo es una mala idea en cualquier ecosistema, y este no es la excepción. Cuando ya tengas unas cuantas instaladas, la auditoría periódica evita que acumules skills rotas, duplicadas o desactualizadas:

```
hermes skills check
```

Resultado observable: tu tiempo por tarea deja de ser lineal. La primera ejecución de un proceso cuesta lo que cuesta; la décima cuesta una línea. Además, y esto se nota mucho en equipos, la calidad se vuelve consistente: la skill no tiene días malos ni se olvida del paso cuatro.

Ejemplo práctico: un estudio de diseño de cinco personas preparaba propuestas comerciales a medida. Entre briefing, alcance, cronograma y presupuesto, cada propuesta consumía entre tres y cuatro horas, y el resultado dependía de quién la escribiera. Convirtieron el proceso en una skill `propuesta-comercial` que toma las notas del briefing y devuelve el documento completo con las seis secciones fijas y los rangos de precio ya calculados según el tipo de proyecto. Bajaron a cincuenta minutos, de los cuales cuarenta son revisión humana y ajuste fino. Con unas nueve propuestas al mes, recuperaron más de veinte horas mensuales. El beneficio que no esperaban: la tasa de aceptación subió, porque la skill nunca se olvidaba de incluir la sección de casos similares, que el equipo omitía la mitad de las veces por prisa.

Nivel 5 — Convertir chequeos recurrentes en crons

Un cron es una tarea que se dispara sola en un horario. Es el punto donde el sistema deja de esperar a que tú te acuerdes. Cualquier cosa que revises «cuando puedo» —métricas de la semana, menciones de marca, facturas pendientes, cambios en la web de un competidor, cola de soporte— es candidato inmediato: son chequeos con periodicidad natural que casi nunca se hacen con la disciplina que merecen.

Lo que distingue un cron útil de uno que acabarás ignorando es el formato de la salida. Un cron que te manda un volcado de datos crudos se convierte en ruido en dos semanas. Un buen cron te dice tres cosas: esto es lo que ha cambiado, esta es la acción que recomiendo, y además, ¿quieres que lo haga por ti? Esa tercera parte es la que convierte una notificación en una decisión que puedes tomar en quince segundos desde el móvil.

La sintaxis acepta cron estándar y también expresiones más legibles como `"30m"` o `"every 2h"`. Y puedes elegir a dónde llega el resultado —`telegram`, `local`, `discord`, `signal` o `platform:chat_id`— con `--deliver`:

```
hermes cron create "0 9 * * 1" "revisa las métricas de la semana pasada, compáralas con
las cuatro anteriores, señala solo las desviaciones superiores al 15 % y propón una acción
por cada una" --name metricas-semanales --deliver telegram --skill informe-metricas

hermes cron create "0 8 * * *" "revisa la cola de soporte, agrupa por motivo y avísame si
algún ticket lleva más de 24 horas sin respuesta" --name cola-soporte --deliver telegram

hermes cron list
hermes cron status
```

La advertencia importa tanto como la técnica: no te pases con los crons. Es la trampa clásica de este nivel. Empiezas con dos, te entusiasmas, y a los tres meses tienes catorce, de los cuales seis vigilan cosas que ya no importan y dos fallan en silencio. Cuando el ruido supera a la señal, dejas de leerlos —incluidos los tres que sí eran valiosos.

La higiene es sencilla y hay que hacerla, digamos, cada trimestre. Mira qué se ha ejecutado y con qué resultado, pausa lo dudoso en lugar de borrarlo, y comprueba a mano una tarea antes de dejarla suelta:

```
hermes cron runs
hermes cron history
hermes cron pause 3
hermes cron resume 3
hermes cron run 7
hermes cron edit 5
```

Criterio de decisión: si en el último mes un cron no ha provocado ni una sola acción por tu parte, o lo reformulas o lo pausas.

Resultado observable: pasas de operación reactiva a operación vigilada. Los problemas se detectan cuando aparecen, no cuando ya han hecho daño.

Ejemplo práctico: una tienda online tenía un problema recurrente con las roturas de stock de sus veinte referencias más vendidas. Lo detectaban cuando un cliente se quejaba, ya tarde. Montaron un cron diario a las 7:30 que cruza el inventario con la velocidad de venta de los últimos catorce días y avisa por Telegram de qué referencias caerán por debajo de siete días de cobertura, con la cantidad de reposición sugerida. En el primer trimestre las roturas pasaron de nueve a una. La tarea que antes «tocaba hacer los viernes» ahora se hace cada día y no la hace nadie.

Nivel 6 — Conectar las fuentes de datos de tu empresa

Hasta el nivel 5 le das el contexto a mano: archivos, datos pegados, exportaciones. El nivel 6 es cuando el agente accede directamente a donde vive la información —analítica, CRM, facturación, documentos, calendario, repositorios— sin que tú hagas de puente.

El error de encuadre habitual es pensar que el valor está en la conexión. No lo está. La magia de los conectores no son los conectores en sí: es la síntesis. Que tu analítica, tu CRM, tu información de ingresos y tus documentos internos puedan hablar entre sí en la misma pregunta es lo que produce respuestas que ninguna de esas herramientas te da por separado. «¿Qué canal trae los leads que más tardan en cerrar, y cuánto nos cuesta esa demora?» es una pregunta que atraviesa tres sistemas. Ninguno de los tres la responde solo.

En Hermes esto se articula por dos vías: los toolsets internos y los servidores MCP, que son el estándar para exponer herramientas y datos externos a un agente.

```
hermes tools list
hermes tools enable filesystem
hermes mcp add
hermes mcp list
```

Y si quieres que Hermes actúe como proveedor de contexto para otro sistema en lugar de solo consumirlo:

```
hermes mcp serve
```

Aquí es donde la seguridad deja de ser teórica. Cada conexión amplía la superficie de lo que el agente puede ver y tocar. Tres criterios antes de conectar nada:

1. **Mínimo privilegio real.** Si con acceso de solo lectura a un subconjunto de datos ya resuelves lo que necesitas resolver, no des acceso completo por comodidad. La comodidad de hoy es el incidente de dentro de seis meses.
2. **Separación por perfil.** No mezcles el entorno donde exploras con el que toca datos de clientes o sistemas de producción. Los perfiles existen exactamente para esto:

```
hermes profile create cliente-acme
hermes profile use cliente-acme
hermes profile list
```

1. **Auditoría antes de escalar.** Antes de conectar una fuente sensible, pasa la revisión y deja constancia del estado:

```
hermes security
hermes verify
hermes backup
```

Y ten localizado el freno de mano. Si algo se comporta de forma rara —una herramienta devuelve datos que no esperabas, un cron entra en bucle— el orden correcto es parar primero e investigar después:

```
hermes pause
hermes checkpoints
hermes resume
```

Resultado observable: desaparece el trabajo de fontanería. Nadie exporta CSVs, nadie copia cifras entre pestañas, y las preguntas que antes requerían media mañana de recopilación se responden en el momento.

Ejemplo práctico: una consultora B2B con veinticinco clientes activos tardaba dos días en cerrar cada revisión trimestral, y la mayor parte del tiempo se iba en recopilar datos de cuatro sistemas distintos. Al conectar las fuentes con acceso de solo lectura y bien acotado, la recopilación pasó a ser automática y los dos días se convirtieron en medio: cuatro horas de análisis y criterio, cero de copiar y pegar. El dato que más les sorprendió llegó de la síntesis, no de la automatización: al cruzar horas facturadas con margen por cliente descubrieron que dos de sus cuentas «grandes» tenían margen negativo desde hacía tres trimestres. Ningún sistema por separado lo mostraba.

Nivel 7 — Añadir permisos y gobernanza

Un agente conectado a los sistemas de tu empresa y capaz de ejecutar tareas solo necesita reglas explícitas sobre qué puede hacer sin preguntar y qué no. Este nivel no es burocracia: es lo que hace que puedas darle más alcance sin perder el sueño. Sin gobernanza, la reacción natural ante el primer susto es cortar todas las conexiones, y ahí pierdes los seis niveles anteriores.

El marco es de tres categorías y es más útil de lo que su simplicidad sugiere.

Lectura. Suele estar bien concederla, siempre que esté bien acotada. Leer no destruye nada. El riesgo real no es el daño sino la exposición: qué información acaba en un contexto donde no debería estar. Por eso «acotada» es la palabra que hace el trabajo. Acceso a la carpeta de informes, no al disco entero. A la analítica, no a la base de datos de producción con los datos personales dentro.

Borradores. Aquí Hermes debe ser agresivo. Que redacte, que proponga, que prepare el correo entero, que genere la propuesta completa, que escriba el borrador del contrato. Un borrador no tiene consecuencias externas: nadie lo recibe, nada se publica, ningún dinero se mueve. Frenar al agente en la fase de borrador es el error más caro de este nivel, porque es exactamente donde aporta más valor por minuto. La mentalidad correcta es «que llegue hasta el último centímetro antes de la puerta».

Escritura externa. Cualquier acción que salga al mundo —enviar el correo, publicar el post, modificar un registro en el CRM, ejecutar un pago, borrar archivos— debe requerir aprobación humana. Es la línea que no se cruza. No porque el agente sea poco fiable, sino porque el coste de un error asimétrico es enorme: revisar un borrador cuesta treinta segundos, retractar un correo enviado a mil quinientos suscriptores cuesta una reputación.

Cuatro decisiones concretas para implementar hoy:

1. **Define tu lista de acciones irreversibles y decláralas de aprobación obligatoria.** Escríbela en un archivo, no la dejes en tu cabeza: enviar comunicaciones externas, publicar en cualquier canal público, modificar o borrar registros en sistemas de negocio, mover dinero, tocar configuración de producción.
2. **Separa perfiles por nivel de riesgo.** Un perfil para exploración y borradores con permisos amplios de lectura, y un perfil distinto y estrecho para lo que toca datos de clientes. `hermes profile create` es el mecanismo; la disciplina de no mezclarlos, tuya.
3. **Revisa qué está habilitado y apaga lo que no uses.** La superficie de riesgo crece por acumulación silenciosa, no por decisiones grandes.

```
hermes tools list
hermes tools disable shell
hermes security
```

1. **Ten un procedimiento de parada y de vuelta atrás, probado antes de necesitarlo.** Un freno que no has usado nunca no es un freno, es una suposición:

```
hermes pause
hermes checkpoints
hermes backup
hermes resume
```

Resultado observable: puedes ampliar el alcance del agente sin ampliar el riesgo. Es la paradoja de este nivel: la gobernanza no limita lo que el sistema hace, es lo que te permite dejarle hacer más.

Ejemplo práctico: una agencia con seis personas empezó con el agente en modo «pregunta antes de cada acción», lo cual era tan lento que la gente dejó de usarlo. Reescribieron las reglas con el marco de tres categorías: lectura libre dentro de las carpetas de proyecto, borradores sin fricción ninguna, y una lista explícita de once acciones que exigen un «sí» humano. El uso se multiplicó por cuatro en un mes, con cero incidentes. La lección: el permiso mal calibrado no genera seguridad, genera abandono —y un agente abandonado tiene un ROI exactamente igual a cero.

Nivel 8 — Cerebro de empresa

El último nivel tiene dos componentes y ninguno funciona sin el otro: **multijugador** y **memoria**.

Multijugador significa sacar el agente de tu terminal y ponerlo donde tu equipo ya conversa. Mientras Hermes viva solo en tu máquina, es tu asistente personal por muy sofisticado que sea. En cuanto vive en un canal de Slack, un grupo de Telegram o un servidor de Discord, se convierte en infraestructura compartida.

```
hermes gateway
hermes slack
hermes whatsapp
hermes send
```

El cambio no es técnico, es cultural, y se nota en un detalle concreto: aparecen las preguntas cruzadas. Alguien de soporte pregunta algo en el canal, y el comercial que lo lee de reojo descubre un patrón de objeciones que no conocía. La respuesta del agente es la mitad del valor; la otra mitad es que ocurra delante del equipo. Como dice Siu, cuanto más rápido intercambiamos ideas, más rápido crecemos como empresa. Un agente en un canal común acelera precisamente ese intercambio, porque el conocimiento deja de estar en la cabeza de quien preguntó.

Memoria es el segundo componente, y es el que separa un asistente de un cerebro. La formulación de Siu es exacta: si un agente no puede recordar sus decisiones, no puede convertirse en un cerebro. Sin memoria, cada sesión empieza de cero y vuelves a explicar quién es tu cliente ideal, qué probasteis en marzo y por qué se descartó. El trabajo se evapora, y evaporarse es la palabra correcta: no se pierde de golpe, se disipa en decisiones que nadie registró.

```
hermes memory setup
hermes memory status
hermes sessions list
hermes sessions export
```

Lo que merece la pena guardar en memoria no es cada conversación —eso es ruido acumulado— sino las decisiones y sus motivos. Qué probasteis, qué funcionó, qué descartasteis y por qué, qué restricciones son inamovibles, quién decide qué. Ese es el sustrato que convierte al agente en alguien que ya conoce la casa.

El dojo de habilidades. Cuando el equipo comparte skills, se produce un efecto que no se ve en el nivel 4. Cada persona aporta procesos de su especialidad a una biblioteca común: comercial sube la skill de cualificación de leads, soporte la de clasificación de incidencias, finanzas la del cierre mensual. El equipo entero tiene acceso al mejor procedimiento de cada área, no solo al suyo. Y como las skills son archivos de texto, se mejoran igual que el código: alguien detecta un fallo en el paso tres, lo corrige, y a partir de ahí toda la empresa ejecuta la versión buena.

Mantener esa biblioteca sana requiere lo mismo que cualquier otra: revisión periódica.

```
hermes skills list
hermes skills check
hermes skills inspect propuesta-comercial
```

Resultado observable: el conocimiento operativo deja de vivir en las cabezas. Una persona nueva es productiva en una semana en lugar de en dos meses, porque los procesos están escritos y ejecutables. Y cuando alguien se va, se va la persona, no el proceso.

Escenario ilustrativo (no es un cliente real; es la forma que suele tomar este nivel): una empresa de servicios de treinta personas con el problema clásico, tres personas concentrando el grueso del conocimiento operativo y las vacaciones de agosto convertidas en un evento de riesgo. En seis meses montaron una biblioteca de treinta y una skills cubriendo sus procesos habituales, con el agente presente en cuatro canales de Slack y memoria activa sobre decisiones de producto y cliente. El indicador que eligieron para medirlo fue el tiempo de incorporación: de siete semanas hasta ser autónomo a doce días. El segundo efecto lo vieron después: la dirección dejó de recibir preguntas de proceso y empezó a recibir preguntas de criterio, que son las que de verdad requieren a un humano.

Multijugador y el cerebro de empresa en la práctica

Merece la pena detenerse en cómo se implanta el nivel 8, porque es donde más intentos fracasan —y casi nunca por motivos técnicos.

El primer error es empezar por todas las superficies a la vez. Un canal, un caso de uso claro, dos semanas de rodaje. Cuando la gente entienda para qué sirve ahí, abre el segundo. Un agente en seis canales sin propósito definido en ninguno se convierte en ruido de fondo que la gente silencia en diez días.

El segundo error es no definir qué se pregunta en público y qué en privado. La regla que funciona: si la respuesta puede ser útil para alguien más, va al canal. Si es una exploración personal o toca datos sensibles, va a tu sesión privada. Sin esa distinción explícita pasa una de dos cosas: o el canal se llena de trivialidades, o nadie se atreve a preguntar nada por miedo a molestar.

El tercer error es tratar la memoria como un archivo pasivo. La memoria hay que curarla: revisar cada cierto tiempo qué se ha registrado, corregir lo que envejeció mal y borrar lo que dejó de ser cierto. Una decisión de hace un año que ya no aplica y sigue en memoria es peor que no tener memoria, porque el agente la usará con total confianza.

Y una nota sobre qué esperar del agente cuando llega a este nivel. El mejor agente no es necesariamente el más obediente: es el que tiene criterio. Uno que ejecuta al pie de la letra una instrucción mal formulada te da exactamente lo que pediste. Uno con buen criterio te dice que la pregunta está mal planteada y propone la que deberías estar haciendo. Ese salto solo ocurre cuando tiene contexto suficiente —memoria, artefactos previos, skills que codifican tu forma de trabajar— para tener una opinión informada sobre tu negocio.

Gobernanza: cómo implantarla sin frenar el sistema

Recapitulando el marco de permisos en forma operativa, porque es la parte que más gente se salta y la que más caro sale saltarse.

Categoría	Postura por defecto	Qué acota el riesgo
Lectura	Permitir	Alcance estrecho: carpetas y fuentes concretas, no acceso total
Borradores	Permitir sin fricción	Ninguna acción sale al exterior; el coste de un mal borrador es cero
Escritura externa	Requiere aprobación humana	Lista explícita y escrita de acciones irreversibles

La razón de implantar esto **antes** de escalar y no después es que el orden inverso no funciona. Si primero conectas las fuentes y das permisos amplios, la primera vez que algo salga mal la reacción será desconectarlo por completo, y volverás al nivel 1 con la convicción añadida de que «esto de los agentes no funciona». Con las reglas puestas desde el principio, un fallo es un incidente acotado del que aprendes, no una crisis de confianza.

Un apunte sobre proporción: la gobernanza debe ser proporcional al alcance. Si trabajas solo y el agente únicamente lee archivos de tu carpeta de proyectos, no necesitas un comité. Si hay cinco personas y el agente toca el CRM, sí necesitas la lista escrita de acciones irreversibles. Escala las reglas al mismo ritmo que escalas los permisos, ni antes ni después.

Y revisa la configuración cada cierto tiempo, porque la deriva es real: se habilitan herramientas para una prueba puntual y nadie las vuelve a apagar.

```
hermes config
hermes doctor
hermes security
hermes tools list
```

Cierre: construye sistemas que compongan

La mayoría está atascada en el nivel uno. No por falta de capacidad técnica, sino porque el nivel uno funciona lo bastante bien como para no obligar a nadie a salir de ahí. Preguntas, obtienes una respuesta útil, cierras. Es un óptimo local perfectamente cómodo, y es exactamente donde se queda sin recoger todo lo que hay de los siete niveles siguientes.

Subir de nivel no requiere un plan de transformación. Requiere un hábito distinto, y se resume en tres movimientos:

Pide el archivo. Cada vez que un resultado tenga valor para alguien más que tú, pide el artefacto. Es una frase más en la instrucción y multiplica los lectores por ocho.

Escribe la skill a la segunda vez. No a la quinta. En cuanto notes que estás reexplicando un proceso que ya explicaste, para y escríbelo en `~/.hermes/skills/<nombre>/SKILL.md`. Los prompts son desechables; las skills componen.

Programa el chequeo. Cualquier cosa que revises «cuando puedo» es un cron esperando a existir. Empieza por uno solo, el que más te duela olvidar, y déjalo correr un mes antes de añadir el segundo.

Con esos tres hábitos, los niveles 6, 7 y 8 llegan casi por gravedad: cuando tienes artefactos, skills y crons funcionando, conectar las fuentes de datos deja de ser un proyecto y pasa a ser el siguiente paso obvio. La gobernanza aparece porque la necesitas de verdad. Y el cerebro de empresa no es una meta que se alcanza un martes: es lo que tienes cuando llevas seis meses acumulando en lugar de consumir.

La diferencia entre alguien en el nivel 1 y alguien en el nivel 8 no es el modelo que usan. Es que uno lleva medio año generando respuestas que se evaporaron y el otro lleva medio año generando activos que siguen ahí. Ese es el objetivo real: no automatizar tareas sueltas, sino construir sistemas reutilizables que compongan. Cada skill hace más valiosa a la siguiente, cada artefacto alimenta el próximo proceso, cada decisión recordada mejora la siguiente. Lo que hoy te ahorra veinte minutos, dentro de un año es la razón por la que puedes operar con la mitad de fricción que tu competencia.

Empieza por el nivel siguiente al tuyo. Solo uno.

Marco inspirado en el vídeo de Eric Siu (Leveling Up with Eric Siu, julio de 2026); el desglose, los ejemplos y los comandos son desarrollo original de esta guía. Documento no afiliado ni patrocinado por Eric Siu, Leveling Up, Single Grain ni Nous Research. Hermes Agent es un producto de Nous Research.