

MCP, API O CLI

TRES CAMINOS, UNA REGLA



TRES CAMINOS DISTINTOS

MCP, API y CLI no son tres nombres para lo mismo. Son tres caminos para que Claude llegue a tus herramientas, y cada uno tiene su lugar.

	QUÉ ES	CUÁNDO BRILLA	LO QUE CUESTA
API	El contrato entre dos programas: si me mandas esto, te devuelvo aquello.	Un producto que tiene que funcionar solo, sin ti ni la IA en medio.	Alguien escribe y mantiene el código que la llama.
CLI	Un programa que manejas con comandos en la terminal: gh, vercel, railway.	Cuando existe una buena. Claude Code vive en la terminal: es su idioma nativo.	Tenerla instalada y con sesión iniciada.
MCP	Un estándar abierto para conectar herramientas a una IA.	Sin CLI, cuando el servicio necesita tu cuenta, o para manejar un sistema tuyo.	Cada servidor es una puerta más, con tus accesos.

Por debajo, casi todo termina llamando a una API. La CLI y el MCP son dos formas distintas de que Claude llegue a ella.



POR QUÉ LA CLI SUELE GANAR

Claude Code trabaja en tu terminal: ya conoce los comandos, los encadena y lee solo la respuesta que necesita.

```
gh pr checks          # qué revisiones fallaron
gh run view --log-failed # el log de la que falló
```

Te basta con pedir: «revisa por qué falló el PR y muéstrame el error». Él elige los comandos.

CUÁNDO SÍ CONVIENE UN MCP

CASO 1

SIN CLI O CON TU CUENTA

Gmail, Calendar o Drive: conectas una vez con tu cuenta y listo.

CASO 2

UN SISTEMA TUYO

Envuelves tu API en un servidor MCP y la manejas pidiéndolo con palabras.

```
claude mcp list          # los servidores configurados
claude mcp remove <nombre> # quitar uno
```

Claude Code carga las herramientas MCP bajo demanda: al inicio solo lee sus nombres y las instrucciones de cada servidor. Igual, cada servidor corre con tus accesos: conecta solo lo que vas a usar.

LA REGLA DE TRES PREGUNTAS

1

¿EXISTE UNA CLI BUENA PARA ESE SERVICIO?

Úsala. Es lo más simple y lo que Claude Code mejor maneja.

2

¿NO EXISTE, O NECESITA TU CUENTA?

Conecta un MCP, de una fuente de confianza y con los permisos mínimos.

3

¿ES UN PRODUCTO QUE FUNCIONA SIN TI?

Eso va por API. Si después quieres manejarlo con IA, le pones un MCP encima.

CHECKLIST

- ☐ Revisé si el servicio tiene una CLI oficial.
- ☐ Tengo la CLI con sesión iniciada en mi terminal.
- ☐ Cada MCP conectado es uno que uso de verdad.
- ☐ Sé qué permisos tiene cada servidor: leer, editar o borrar.
- ☐ Las claves de mis APIs viven en variables de entorno, no en el código.
- ☐ Leo cada acción antes de aprobarla.

UN MODELO PARA CADA TAREA

Claude Code reparte tareas entre subagentes, y cada uno puede usar un modelo distinto. Si no indicas ninguno, el subagente hereda el de tu conversación.

MODELO	PARA QUÉ
HAIKU	Buscar archivos, leer, renombrar, ordenar.
SONNET	Implementar lógica y escribir tests.
OPUS	Decidir arquitectura y revisar seguridad.

Tres formas de fijarlo: en el pedido («lanza un agente con haiku para...»), con el campo `model` de un archivo en `.claude/agents/`, o para todos con la variable `CLAUDE_CODE_SUBAGENT_MODEL`.

```
Revisa qué servidores MCP tengo conectados (claude mcp list) y qué CLI tengo
instaladas. Para cada servicio, dime si conviene la CLI o el MCP y por qué.
Después créame tres subagentes en .claude/agents/: uno con model: haiku para
buscar archivos, uno con model: sonnet para implementar y uno con model: opus para
revisar decisiones de arquitectura.
```

Y ojo: delegar no es desentenderse. Tú decides qué se construye y revisas lo que vuelve.

Sígueme en [Elkis Dev](#) para más IA y desarrollo, explicados simple.