Día 14 · Tema 5 — Subagentes, MCP y human-in-the-loop

Eduardo C. Garrido-Merchán · ecgarrido@comillas.edu Universidad Pontificia Comillas · ICADE

1. Subagentes: la jerarquía

Una sesión compleja excede la capacidad de un único agente: el contexto se satura, las decisiones se mezclan, la traza se vuelve ilegible. La solución es invocar *subagentes* especializados:

$$\pi_{\text{coord}}(c_t) \;\to\; \text{spawn}(\pi_{\text{sub}}, \text{subtask}) \;\to\; (\text{subagent ejecuta}) \;\to\; \text{resultado al coordinador.} \tag{1}$$

El subagente tiene *su propio contexto* c_0' que sólo contiene la subtarea y las herramientas relevantes. Al terminar devuelve un resumen ejecutivo, no la traza entera. El coordinador conserva contexto limpio; los detalles se procesan abajo.

2. Cuándo usar subagentes

- **Búsqueda amplia**: explorar una base de código grande sin ensuciar el contexto principal.
- **Tareas heterogéneas paralelas**: implementación + tests + documentación, cada uno con un subagente.
- **Acceso a tools distintos**: el subagente *security* tiene acceso a `gh` y `sast`; el subagente *ux* a la herramienta de browser. Separar evita ruido.

Anti-patrón: spawn-on-everything. Cada subagente cuesta tokens (su contexto se carga; su respuesta se vuelve a leer). Si la subtarea cabe en 3 turnos del agente principal, no se delega.

3. MCP: Model Context Protocol

MCP es un protocolo estándar (JSON-RPC sobre stdio o HTTP) para conectar un LLM con *servidores* externos que exponen tools, recursos y prompts. La firma de una tool MCP es:

$$\text{tool} := \big(\,\text{name},\;\text{description},\;\text{input\_schema},\;\text{output\_schema}\,\big). \tag{2}$$

El cliente (Claude Code, Cursor, ChatGPT desktop) descubre los tools disponibles llamando a `tools/list`, invoca con `tools/call`, recibe el resultado y lo presenta al modelo. El modelo no sabe nada de la implementación: ve sólo el contrato JSON.

4. Por qué MCP cambia las reglas

Antes de MCP, cada cliente reimplementaba conectores específicos (Slack, GitHub, Salesforce…). MCP estandariza: el equipo que mantiene un servidor lo escribe una vez y cualquier cliente compatible lo consume. El catálogo de servidores MCP en 2026 cubre filesystem, GitHub, Postgres, Notion, Slack, Linear, Sentry, etc.

5. Tool use a bajo nivel: JSON estructurado

El LLM emite una llamada con la forma

```
{ "name": "search_crm",
  "args": { "customer_id": "C-1234" } }
```

y el servidor responde

```
{ "name": "Acme S.A.", "tier": "gold", "open_tickets": 2 }
```

Ambos siguen un *JSON Schema* explícito. Validar las entradas y salidas con `pydantic` o `zod` es la forma estándar de que un agente no se convierta en una caja negra.

6. Human-in-the-loop (HITL)

Para acciones irreversibles (borrar tablas, enviar emails, transferir dinero) la política se convierte en

$$\pi_\theta(c_t) \;=\; \begin{cases} \text{ejecutar } a_t & \text{si } \text{risk}(a_t) \le \rho, \\ \text{pedir confirmación al humano} & \text{si } \text{risk}(a_t) > \rho. \end{cases} \tag{3}$$

$\text{risk}(a_t)$ score de riesgo del action (heurístico o aprendido)
ρ umbral configurable; típicamente bajo en producción

En la práctica, Claude Code aplica HITL automáticamente a ciertas operaciones (push a main, drop de tabla) y deja pasar otras.

7. Caso comercial: agente de soporte tier-1

Una telco recibe 10^4 consultas/día. El 60 % son repetitivas (cambiar tarifa, consultar factura). Pipeline:

1. **Triage** (agente principal): clasifica intent y decide.
2. **Subagente CRM** con MCP a Salesforce: lee datos del cliente.
3. **Subagente facturación** con MCP a billing system.
4. Si la respuesta es de riesgo bajo, ejecuta y notifica al cliente.
5. Si el riesgo es alto (cambio de tarifa, baja), escalado a humano.

Idea clave. El humano deja de procesar las consultas triviales y sólo ve las que el agente marca. Telefónica reportó un −40 % de tiempo medio de resolución con este patrón.

8. Métricas que debe registrar el agente

Para auditoría y mejora:

- Tasa de éxito (porcentaje de tareas completadas sin escalado).
- Coste medio por tarea ($).
- Latencia (segundos hasta resolución).
- Tasa de *tool errors* (llamadas con argumentos inválidos).
- Tasa de *hallucination* (afirmaciones contradichas por la traza).