Agent Chaining: Encadenamiento Automático entre Agentes de Redacción y Marketing

Estantería: DevSecOps e Infraestructura

Subtema: Gobernanza

mercedev.es — 2026-05-19 | Epic 3 - Fase 2

El Desafío (Síntoma)

Se detectó una fricción operativa significativa en el flujo de creación de contenido. Aunque el Agente Bibliotecario automatizaba la redacción técnica estructurada, el proceso posterior de promoción estaba fragmentado.

Era necesario invocar manualmente al Agente Blogger en una sesión separada para generar el contenido de marketing social (DevRel). Adicionalmente, el Blogger operaba a ciegas respecto a la ubicación final del documento, lo que imposibilitaba inyectar automáticamente la URL canónica (destino del post promocional) y aumentaba el riesgo de enlaces rotos o desincronizados.

La Maniobra (Lógica)

Se implementó el patrón de Agent Chaining (Encadenamiento de Agentes) mediante orquestación en Python.

La arquitectura se modificó para que el Agente Bibliotecario actúe como nodo de origen. Al finalizar la generación y validación del documento técnico, el script evalúa el estado del proceso y solicita confirmación interactiva. Si se aprueba la promoción, el Bibliotecario invoca directamente al Agente Blogger pasándole la ruta absoluta del archivo recién creado como argumento. A su vez, el Blogger fue refactorizado para ingerir este contexto: extrae el YAML Frontmatter del archivo entrante, aplica una función de *slugify* al título y deduce matemáticamente cuál será la URL final del documento estático. Esta URL se inyecta dinámicamente como directiva en el prompt del modelo local, forzando a la IA a incluirla en el bloque de anuncio.

El Aprendizaje / Deuda Técnica

- **Separation of Concerns en Modelos Locales:** Forzar a un único SLM (Small Language Model - Modelo de Lenguaje Pequeño) a ser ingeniero técnico y redactor publicitario simultáneamente provoca alucinaciones (colapso de modo). Separarlos en dos agentes de propósito único garantiza calidad extrema en ambos frentes.
- **Fricción Cero mediante Context Passing:** Encadenar agentes locales reduce la fricción operativa a cero. Cada IA hace solo una tarea bien definida, pasándose el contexto exacto y el control de forma programática mediante Python sin requerir interacciones manuales de extracción e inyección de datos.

En resumen

Anteriormente, al finalizar la redacción de un manual técnico con la primera Inteligencia Artificial, era necesario abrir una herramienta distinta, ordenar la escritura de un anuncio para redes sociales y buscar manualmente el enlace final de la página para pegarlo en el texto. Con la nueva arquitectura, se ha creado una "cadena de montaje": en cuanto la primera IA termina su trabajo técnico, le pasa automáticamente el relevo a la segunda IA (la de marketing), calculando e inyectando por sí misma el enlace definitivo para que el anuncio se guarde perfectamente conectado, eliminando cualquier intervención humana intermedia.