

Manual de Instalacion Directrices para Sistemas Agenticos

Sistema multi-agente para flujos de trabajo con IA

Version 1.0 - Mayo 2026

Compatible con Claude Code Google CLI Open Code

Brenno Henrique Pereira dos Santos

Creador y autor del sistema



Contenido

01	Que son las directrices?	2
02	Ventajas de un sistema agentico	3
03	Esta es una base: personaliza o inspirete	5
04	Requisitos previos	6
05	Guia de instalacion	7
06	Flujograma de trabajo	9
07	Agentes disponibles	10
08	Skills disponibles	12
09	Como usar las directrices	13
10	Estructura de carpetas y FAQ	15

01 | ¿Qué son las directrices?

Las **directrices para sistemas agénticos** son un conjunto de instrucciones y protocolos diseñados para guiar agentes de IA (como Claude) en flujos de trabajo de proyectos complejos. Funcionan como un sistema operativo de toma de decisiones: enrutan solicitudes, coordinan especialistas y garantizan consistencia en cada entrega.

Características principales

Característica	Descripción
Sistema multi-agente	Especialistas con roles definidos (Arquitecto, Dev, UX, Diseñador, etc.)
Protocolos iterativos	Flujos de evaluación consensuada entre expertos antes de ejecutar
Enrutamiento automático	La solicitud llega al agente o protocolo correcto sin intervención
Personalización	Compatible con Claude Code, Google CLI, Open Code y otros CLIs
Escalabilidad	Desde proyectos pequeños hasta enterprise

¿Para quién es?

- Equipos de desarrollo que trabajan con asistentes de IA
- Proyectos que necesitan estructura y consistencia en decisiones técnicas
- Organizaciones que quieren un flujo de trabajo estandarizado
- Desarrolladores que usan Claude Code, Google CLI u otras herramientas similares

02 | Ventajas de un sistema agéntico

Trabajar con un sistema de agentes especializados no es solo una cuestión de organización: es un cambio de paradigma en cómo se toman decisiones técnicas y se ejecutan proyectos con asistentes de IA. Estos son los beneficios concretos frente a usar un solo agente genérico.

Consistencia en las decisiones

Un agente genérico responde diferente cada vez que le preguntas lo mismo. Un sistema con roles definidos aplica siempre los mismos criterios: el Arquitecto evalúa escalabilidad, el Dev evalúa implementabilidad, el UX evalúa la experiencia. El resultado es predecible y trazable.

División de responsabilidades clara

Cada agente tiene un dominio de conocimiento acotado. Esto reduce el riesgo de respuestas superficiales que mezclan criterios contradictorios. Cuando el Diseñador opina sobre un componente, no interfiere con la lógica que evalúa el Dev, y viceversa.

Evaluación multi-ángulo antes de ejecutar

El Protocolo CONSEJO convoca especialistas antes de tomar decisiones críticas. En lugar de una respuesta única que puede estar sesgada, obtienes una evaluación consensuada desde arquitectura, implementación, experiencia de usuario y diseño, con riesgos y tradeoffs explicitados.

Escalabilidad del flujo de trabajo

A medida que el proyecto crece, el sistema escala con él: se añaden agentes especializados (Seguridad, Mobile, Data) que se activan solo cuando son relevantes. No hay sobrecarga cognitiva de un agente intentando cubrir todo.

Enrutamiento automático sin fricción

El usuario no necesita saber qué agente invocar. El sistema clasifica la solicitud y la dirige al protocolo correcto: una pregunta simple va directo a respuesta, una feature compleja pasa por ANALISTA → PLANNER → EXECUTOR sin que el usuario lo gestione manualmente.

Trazabilidad y auditoría del proceso

Al tener agentes con roles diferenciados, es posible entender por qué se tomó cada decisión: el Arquitecto recomienda X por escalabilidad, el Dev advierte sobre Y por complejidad de implementación. El razonamiento queda explícito, no enterrado en una respuesta monolítica.

Reducción de errores en cambios críticos

El DEBUGUER se activa automáticamente en cambios de alto riesgo (Nivel 5) y el VIGILANTE en todo lo que toca autenticación, pagos o datos sensibles. Esta capa de verificación sistemática reduce la probabilidad de introducir vulnerabilidades o regresiones sin detección.

Contexto persistente del proyecto

Los skills /mapa-proyecto y /contexto-proyecto mantienen documentacion viva del estado actual del sistema. Cualquier agente puede consultar ese contexto para tomar decisiones informadas, incluso en sesiones nuevas donde el historial de conversacion no esta disponible.

03 | Esta es una base: personaliza o inspirete

Una base abierta, no un sistema cerrado

Este sistema fue diseñado por **Brenno Henrique Pereira dos Santos** como una arquitectura funcional y probada para trabajar con agentes de IA en proyectos reales. Su intención no es que sea adoptado tal cual, sino que sirva como punto de partida sólido desde el que cada equipo o profesional construya su propia versión.

Que puedes hacer con este sistema

Usarlo directamente	Instala las directrices en tu CLI y empieza a trabajar con el sistema tal como esta. Funciona desde el primer día para equipos de desarrollo que necesitan estructura y consistencia sin tener que diseñar el sistema desde cero.
Adaptarlo a tu contexto	Modifica los agentes existentes para que reflejen los roles de tu equipo. Cambia los criterios del Arquitecto, ajusta los triggers del DEBUGUER, renueva los skills con los flujos específicos de tu organización.
Añadir agentes propios	El sistema está diseñado para ser extensible. Puedes incorporar agentes nuevos —Legal, QA, Marketing, Data Science— siguiendo el mismo patrón de rol + responsabilidad + trigger + métodos.
Usarlo como inspiración	Si prefieres construir tu propio sistema desde cero, esta arquitectura documenta patrones que funcionan: enrutamiento por tipo de solicitud, protocolos iterativos con rondas de consenso, triggers automáticos, skills especializados. Toma lo que te sea útil.

El código fuente de las directrices está disponible para uso educativo y profesional. No hay restricciones para adaptarlo, publicarlo o distribuirlo. Si construyes algo a partir de este sistema, la única intención es que te sea útil.

04 | Requisitos previos

Requisito	Detalle
CLI instalado	Claude Code, Google CLI, Open Code u otra herramienta similar
Acceso a configuración	Capacidad de editar archivos de configuración en tu máquina
Espacio en disco	~5 MB para almacenar las directrices

Modelos de IA recomendados

Modelo	Caso de uso	Prioridad
Claude Opus 4.7	Análisis profundo, tareas complejas, CONSEJO	Alta complejidad
Claude Sonnet 4.6	Implementación, balance velocidad–calidad	Uso general
Claude Haiku 4.5	Tareas rápidas y simples	Alta velocidad

05 | Guía de instalación

Paso 1 — Descargar las directrices

La carpeta **Configuración/** contiene el archivo principal:

`DIRECTRICES_SISTEMA_AGENTICO.md` — Contiene todas las reglas, agentes y protocolos del sistema.

Paso 2 — Configurar en tu CLI

Selecciona tu herramienta y sigue los pasos correspondientes:

Claude Code (recomendado)

1	Abre Claude Code
2	Ve a Configuración → Settings
3	Busca "System Prompt" o "Custom Instructions"
4	Copia el contenido de <code>DIRECTRICES_SISTEMA_AGENTICO.md</code> y pégalo ahí
5	Guarda los cambios

Google CLI (Gemini)

1	Abre tu terminal
2	Navega a la carpeta de Google CLI
3	Busca el archivo <code>config.json</code> (<code>~/google-ai-cli/config.json</code>)
4	Añade una entrada bajo <code>system_prompts</code> con el contenido del archivo
5	Al usar CLI, especifica el prompt: <code>--system-prompt directrices_sistema_agentico</code>

Open Code o CLI personalizado

1	Crea un archivo llamado <code>system-prompt.txt</code> en tu carpeta de configuración
2	Pega el contenido de <code>DIRECTRICES_SISTEMA_AGENTICO.md</code>
3	Configura tu CLI para leer este archivo como prompt del sistema
4	Usa según la documentación de tu CLI específico

Configuración de modelos por agente (opcional)

Agente	Modelo recomendado	Razón
CONSEJO	Opus 4.7	Análisis multi-ángulo, consenso
PLANNER	Opus 4.7	Planes complejos, arquitectura

Agente	Modelo recomendado	Razón
EXECUTOR	Sonnet 4.6	Implementación rápida y correcta
DEBUGUER	Opus 4.7	Verificación profunda, diagnóstico
ACUARIANO	Opus 4.7	Análisis de innovación, segundo orden
Tareas simples	Haiku 4.5	Velocidad en tareas rápidas

Paso 3 — Verificar la instalación

Haz una prueba simple enviando este mensaje a tu agente:

Hola, ¿estás configurado con las directrices agénticas?

Respuesta esperada:

- Responde en español
- Menciona si detecta un proyecto activo
- Está listo para comandos como /mapa-proyecto, /contexto-proyecto, etc.

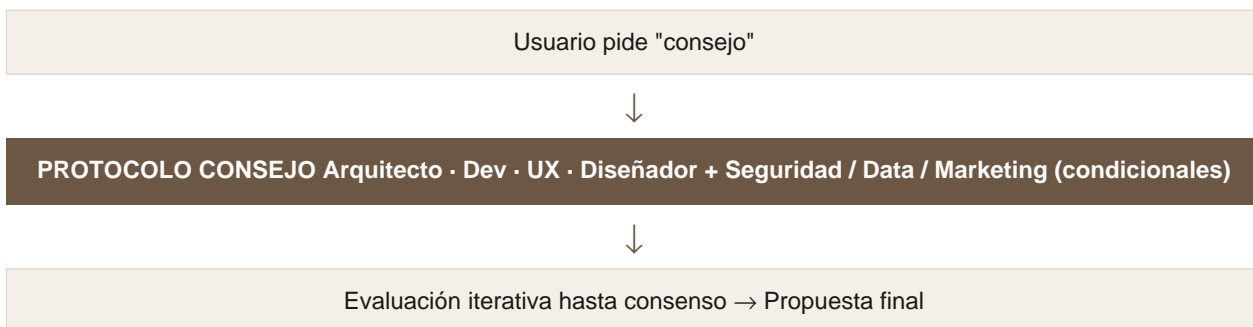
06 | Flujograma de trabajo

Este es el flujo de decisión automático del sistema. Cada solicitud es clasificada y enrutada al agente o protocolo más adecuado sin intervención del usuario.

Flujo principal



Protocolo CONSEJO (multi-agente)



07 | Agentes disponibles

Cada agente es un especialista con rol y responsabilidades definidas. El sistema enruta automáticamente según el tipo de solicitud.

■ CONSEJO		Meta-agente de coordinación
Cuándo	"consejo", "evalúa desde ángulos", "es viable"	
Hace	Convoca especialistas para evaluación multi-ángulo consensuada	
Métodos	Rondas iterativas hasta consenso; especialistas core + condicionales	
■ ARQUITECTO		Especialista de estructura
Cuándo	"ARQUITECTO: ¿es viable X?" o desde CONSEJO	
Hace	Evalúa stack, patrones, escalabilidad y viabilidad técnica	
Métodos	Análisis CLA, matriz de precisión, efectos en cadena	
■ DEV		Especialista de implementación
Cuándo	Estimación de esfuerzo, revisión de arquitectura de código	
Hace	Evalúa código, complejidad, herramientas y viabilidad	
Métodos	Verificación de implementabilidad en tiempo estimado	
■ UX		Especialista de flujo usuario
Cuándo	"UX revisa el flujo", decisiones sobre interfaz de usuario	
Hace	Evalúa flujos, accesibilidad, feedback y experiencia	
Métodos	Evaluación WCAG, micro-interacciones, flujos de usuario	
■ DISEÑADOR		Especialista visual
Cuándo	Componentes UI, revisión visual, design system	
Hace	Evalúa consistencia visual, sistemas de diseño y estética	
Métodos	Design tokens, componentes, coherencia visual	
■ DATA		Especialista de datos
Cuándo	Cambios en BD, nuevas tablas, tracking, privacidad	
Hace	Evalúa modelos de datos, métricas y cumplimiento	
Métodos	Análisis de impacto en datos, privacidad y consistencia	

■ VIGILANTE		Especialista de seguridad
Cuándo	Auth, pagos, datos sensibles (auto-trigger)	
Hace	Evalúa vectores de ataque, vulnerabilidades y riesgos	
Métodos	OWASP, análisis de superficie de ataque	
■ MOBILE		Especialista móvil
Cuándo	Componentes mobile, optimización para dispositivos	
Hace	Evalúa rendimiento, gestos y UX nativa móvil	
Métodos	Optimización de rendimiento, patrones nativos	
■ PLANNER		Planificador de features
Cuándo	Cambio >4 pasos o feature nueva compleja	
Hace	Crea plan explícito con archivos, pasos y verificación	
Métodos	Descomposición en story cards, estimación de impacto	
■ EXECUTOR		Ejecutor de cambios
Cuándo	Plan claro, cambios específicos en una sola área	
Hace	Implementa según plan o story cards	
Métodos	EXECUTOR[DEVOPS] para infraestructura	
■ DEBUGUER		Verificador de cambios
Cuándo	"revisa", "verifica" o auto-trigger plan >5 pasos	
Hace	Verifica comportamiento real, no solo el código	
Métodos	Obligatorio en Nivel 5 (cambios críticos)	
■ ACUARIANO		Especialista de innovación
Cuándo	"Acuarioanaliza el proyecto", evaluación de alternativas	
Hace	Análisis profundo para mejora y perfección	
Métodos	CLA 4 capas, matriz de precisión, análisis en cadena	

08 | Skills disponibles

Los skills son comandos especializados que activan flujos específicos. Se invocan explícitamente o de forma automática según el contexto.

Skill	Tipo	Cuándo usar
/mapa-proyecto	Generación	Usuario dice "crea mapa" o auto
/contexto-proyecto	Generación	Usuario dice "contexto" o auto
/consejo	Protocolo	Usuario pide "consejo"
/consejo-redactor	Protocolo	Usuario pide "redacta/escribe"
/actualizar-mapa	Mantenimiento	Executor tras nueva carpeta/módulo
/actualizar-contexto	Mantenimiento	Executor tras nueva dependencia/patrón
/investigacion-profunda	Búsqueda	Acuario con >5 alternativas
/loop	Automatización	Usuario para tareas recurrentes
/schedule	Automatización	Usuario para cronjobs
/impeccable	Auditoría	Diseñador antes de entregar UI
/vigilancia-completa	Seguridad	Cambios críticos de seguridad
/actualizar-design-system	Mantenimiento	Tras cambios visuales

Flujo /consejo-redactor

ANALISTA SEO	silencioso	Define keywords e intención de búsqueda
REDACTOR	silencioso	Crea texto creativo sin estructura genérica
DETECTOR IA	silencioso · 2 pasadas	Verifica patrones IA, em-dashes, naturalidad
entrega final		

09 | Como usar las directrices

Uso básico

1	Describe con claridad qué necesitas
2	El sistema enruta automáticamente al agente o protocolo correcto
3	Sigue el flujo (puede haber múltiples rondas si la solicitud es compleja)
4	Recibe el resultado: propuesta, plan, código, texto, etc.

Ejemplos de solicitudes

Refactorizar autenticación		
"Necesito refactorizar la autenticación del app"	→	PLANNER → EXECUTOR
Decisión de stack		
"¿Deberíamos usar React o Vue para este proyecto?"	→	CONSEJO (ARQUITECTO + DEV + DISEÑADOR)
Contenido SEO		
"Redacta una descripción SEO para nuestro producto"	→	CONSEJO REDACTOR → texto final
Impacto de BD		
"Agregué una tabla nueva a BD, ¿hay impacto?"	→	DATA (desde CONSEJO si es complejo)
Componente UI		
"Crea un componente de login responsivo"	→	EXECUTOR (+ DISEÑADOR si requiere UI)

Triggers automáticos

Estos eventos activan agentes de forma automática sin que el usuario los invoque:

Evento	Agente	Razón
Usuario dice "consejo"	CONSEJO	Evaluación multi-ángulo
Feature completamente nueva	ANALISTA	Brief formalizado
Cambio >~10 archivos	PLANNER	Plan explícito requerido
Plan >4 pasos	DESCOMPONEDOR	Contexto se pierde entre pasos

Evento	Agente	Razón
Auth / Pagos / Datos sensibles	VIGILANTE	Evaluación de riesgos
Nueva dependencia/patrón	DATA + PROJECT_CONTEXT	Documentación actualizada
Nivel 5 (cambio crítico)	DEBUGUER	Verificación obligatoria
Componente mobile	MOBILE	Optimización para dispositivos

10 | Estructura de carpetas y FAQ

Estructura de carpetas

```
Directrices/  
■■■ README.md  
■ ■■■ Guía de instalación, agentes, skills, uso  
■  
■■■ Configuración/  
■■■ DIRECTRICES_SISTEMA_AGENTICO.md  
■■■ Archivo principal (copiar a tu CLI/editor)
```

- **Para instalar:** Copia el contenido de DIRECTRICES_SISTEMA_AGENTICO.md a tu CLI
- **Para referencia:** Abre README.md para entender agentes y flujo
- **Para mantener:** El versionado es automático — no editar desde README

Preguntas frecuentes

¿Puedo personalizar las directrices?

Sí. Haz cambios locales en tu configuración, sincroniza manualmente si usas múltiples máquinas e incrementa la versión del archivo editado.

¿Funciona con otros modelos además de Claude?

Parcialmente. Gemini es compatible (requiere mapeo de modelos). Open Code es compatible (requiere configuración de skills). Otros LLMs son compatibles pero sin algunas features avanzadas.

¿Qué pasa si mis directrices locales difieren de la raíz?

Las directrices de proyecto (/CLAUDE.md) sobrescriben la raíz (~/.claude/CLAUDE.md). Esto permite overrides locales sin afectar otros proyectos.

¿Necesito usar todos los agentes?

No. Proyecto simple → solo EXECUTOR. Decisión arquitectónica → CONSEJO + PLANNER. Feature compleja → flujo completo (ANALISTA → PLANNER → EXECUTOR → DEBUGUER).

¿Puedo cambiar los modelos por agente?

Sí, si tu CLI lo permite. Tareas complejas → Opus 4.7. Implementación → Sonnet 4.6. Tareas simples → Haiku 4.5.

Brenno Henrique Pereira dos Santos

Creador y autor del sistema agéntico

Version

1.0

Fecha

Mayo 2026

Licencia

Uso educativo y profesional