

Notas

SPAD significa Structured Prompt-Driven Engineering: ingeniería estructurada guiada por instrucciones. Es la metodología de Fernando García Varela para gestionar el desarrollo de software cuando se hace con ayuda de inteligencia artificial.

SPAD no es una colección de instrucciones para pedir código a una IA. Es una forma de trabajar con tres propiedades:

- Secuencial: el trabajo avanza por fases en un orden fijo. Primero se entiende el problema, después se diseña, se revisa el diseño, se definen las pruebas, se construye, se revisa lo construido y se versiona.
- Bloqueante: ninguna fase empieza hasta que la anterior ha producido su resultado y alguien lo ha evaluado. Si una revisión dice que no, se vuelve atrás.
- Auditable: cada fase deja un documento explícito (plan, informe de revisión, estrategia de pruebas, registro de versión) que se guarda y es la entrada de la siguiente.

La lámina muestra la idea con tres pilares: pensar (el diseño es explícito antes de escribir una línea de código), validar (una revisión independiente) y ejecutar (la IA genera código siguiendo reglas, sin tomar decisiones de diseño). Los cimientos son la documentación, la seguridad y la trazabilidad.

Por qué importa a un directivo: cada vez más software de su compañía lo escribe, en parte, una IA. Usted no necesita saber programar para exigir que ese trabajo tenga un método, que alguien responda de él y que deje evidencias.

SPAD es abierto: contenidos con licencia CC BY 4.0 y código con licencia MIT. No existe certificación de SPAD.

Pregunta para el aula: en su organización, ¿quién decide hoy cómo se diseña el software que genera una IA?

TÉRMINOS DE ESTA SLIDE

- **Principio de ejecución bloqueante**: Regla central de SPAD: ninguna fase empieza hasta que la anterior ha entregado su artefacto y ha recibido un veredicto favorable. Evita que la IA construya sobre un plan que nadie ha revisado. El enlace abre el principio en la guía operativa. https://seachad.github.io/AI_CONSULTING/SPAD/html/es/01_SPAD_Guia_operativa.html#el-principio-de-ejecucion-bloqueante

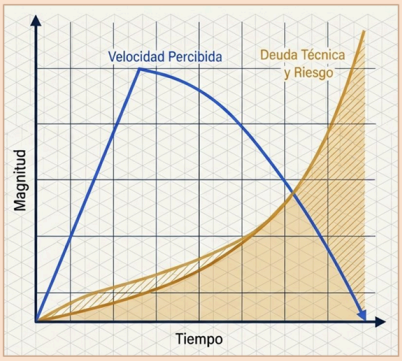
FUENTES Y ENLACES

- **SPAD 00 · Qué es SPAD y para qué sirve** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/00_SPAD_Que_es_y_para_que_sirve.html#que-es-spad
- **SPAD 00 §11 · Estado, licencia y próximos pasos** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/00_SPAD_Que_es_y_para_que_sirve.html#estado-licencia-y-proximos-pasos
- **Qué es SPAD** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/00_SPAD_Que_es_y_para_que_sirve.html
- **Biblioteca SPAD** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/index.html

SPAD EN 10 SLIDES

El problema: la ilusión de velocidad

Gráfica conceptual, sin datos: sin estructura, la velocidad percibida cae mientras la deuda técnica y el riesgo crecen.



- **Al principio** La IA genera código plausible muy rápido. Parece que el equipo vuela.
- **Después** Aparecen decisiones que nadie tomó, pocas pruebas y vulnerabilidades silenciosas: retrabajo.
- **La causa** No es la IA: es la falta de método, que la IA amplifica.
- SPAD mantiene la velocidad donde aporta y pone control donde hace falta.

Detectar un error en la revisión del plan cuesta una conversación; en producción, un incidente.

→ El problema que resuelve

SPAD · Versión 0.x, en revisión · CC BY 4.0 · Fernando García Varela seachad.github.io/AI_CONSULTING/ 2 / 10

Notas

La gráfica es conceptual: no lleva datos. Muestra dos curvas. La velocidad percibida sube muy deprisa al principio, porque la IA escribe código plausible en segundos. Con el tiempo cae, mientras la deuda técnica y el riesgo crecen.

Qué hay detrás, según la página de SPAD:

- El código funciona en la demostración y falla en producción, porque nadie diseñó la solución: se fue pidiendo código hasta que «funcionó».
- Nadie sabe por qué el sistema está construido así, porque las decisiones de diseño las tomó la IA dentro del código.
- Las pruebas son escasas o solo prueban el caso fácil, porque se escriben al final, si sobra tiempo.
- Aparecen vulnerabilidades que nadie revisó, porque la revisión la hizo la misma IA que escribió el código, o no se hizo.
- Una auditoría pide evidencias del desarrollo y no existen, porque el trabajo quedó en conversaciones con la IA.

Ejemplo sencillo: un equipo construye en una semana un portal interno con ayuda de una IA. A los tres meses, un cambio pequeño rompe otra parte del sistema y nadie sabe por qué el portal se hizo así. La semana ganada se pierde en correcciones.

Por qué importa a un directivo: ninguno de estos problemas es de la IA. Son problemas de método que la IA amplifica. Con un método, la calidad no depende de la habilidad de quien escribe las instrucciones ni del modelo del momento, sino de un proceso que cualquier equipo puede repetir.

SPAD no promete cifras de mejora. Al principio hace el trabajo más lento, mientras el equipo aprende. La hipótesis es que reduce el retrabajo y los incidentes; cada organización debe medirlo frente a su punto de partida.

Pregunta para el aula: ¿qué parte de la «velocidad» de su equipo es velocidad real y qué parte es deuda que pagará después?

TÉRMINOS DE ESTA SLIDE

- **SPAD**: SPAD (Structured Prompt-Driven Engineering) es una metodología para construir software con ayuda de la IA de forma ordenada: fases secuenciales, cada una con su artefacto, revisiones con veredicto y personas que validan. Es secuencial, bloqueante y auditable. La usan equipos de desarrollo. El enlace abre su documento de presentación. https://seachad.github.io/AI_CONSULTING/SPAD/html/es/00_SPAD_Que_es_y_para_que_sirve.html

FUENTES Y ENLACES

- **SPAD 00 §2 · El problema que resuelve** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/00_SPAD_Que_es_y_para_que_sirve.html#el-problema-que-resuelve
- **SPAD 00 §10 · Cuándo usar SPAD** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/00_SPAD_Que_es_y_para_que_sirve.html#cuando-usar-spad
- **El problema que resuelve** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/00_SPAD_Que_es_y_para_que_sirve.html#el-problema-que-resuelve



Notas

Esta comparación resume la lámina «IA tradicional frente a SPAD» del sitio. Cada fila es una cuestión que cualquier responsable puede preguntar a su equipo.

- Planificación. Sin método, se pide código hasta que parece funcionar. Con SPAD, el plan es obligatorio, explícito y lo revisa una IA distinta antes de escribir código.
- Pruebas. Sin método, se escriben al final. Con SPAD, la estrategia de pruebas y la cobertura mínima se fijan antes de construir; después una revisión compara la cobertura real con la exigida.
- Revisión. Sin método, la IA se revisa a sí misma. Con SPAD, quien revisa nunca es quien construye y, además, una persona revisa el código.
- Seguridad. Sin método, los fallos se ven en producción. Con SPAD, el código que maneja datos sensibles, autenticación o pagos pasa una revisión de seguridad; los hallazgos críticos y altos bloquean el despliegue.
- Trazabilidad. Sin método, el trabajo queda en conversaciones. Con SPAD, cada fase deja un documento guardado en la carpeta de su tema.

Cuándo usarlo: la página de SPAD lo recomienda mucho en sistemas en producción con clientes, datos sensibles, dinero o decisiones sobre personas, y en sectores con exigencias de auditoría (finanzas, seguros, salud, sector público). Es útil, en versión reducida, en herramientas internas con impacto. Es innecesario en prototipos desechables o programas de uso personal: ahí el coste del proceso supera su beneficio.

Por qué importa a un directivo: le da una lista de preguntas sencillas para saber si su equipo construye con método o confía en la suerte.

Pregunta para el aula: elija un proyecto que conozca. ¿En cuál de las dos columnas estaría hoy?

TÉRMINOS DE ESTA SLIDE

- **Ciclo principal de SPAD** (fases 0–10): Las once fases de SPAD: contexto, plan, revisión del plan, guía de código, estrategia de pruebas, implementación, pruebas, revisión de pruebas, revisión del código, correcciones y gestión de versiones. Cada una deja un artefacto que se revisa. El enlace abre el ciclo en la guía operativa. https://seachad.github.io/AI_CONSULTING/SPAD/html/es/01_SPAD_Guia_operativa.html#ciclo-principal-fases-0-a-10
- **Temas de SPAD**: Un tema es la unidad de trabajo de SPAD: una funcionalidad o un cambio que recorre el ciclo completo con su propia carpeta de artefactos. Permite seguir de principio a fin qué se pidió, qué se planificó, qué se construyó y quién lo validó. El enlace abre su definición. https://seachad.github.io/AI_CONSULTING/SPAD/html/es/02_SPAD_Contextos_temas_y_registro.html#temas

FUENTES Y ENLACES

- **SPAD 00 §2 · El problema que resuelve** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/00_SPAD_Que_es_y_para_que_sirve.html#el-problema-que-resuelve
- **SPAD 00 §10 · Cuándo usar SPAD** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/00_SPAD_Que_es_y_para_que_sirve.html#cuando-usar-spad
- **El problema que resuelve** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/00_SPAD_Que_es_y_para_que_sirve.html#el-problema-que-resuelve
- **Cuándo usar SPAD** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/00_SPAD_Que_es_y_para_que_sirve.html#cuando-usar-spad



Notas

SPAD se apoya en tres verbos. Pensar: el diseño se decide y se escribe antes del código. Validar: una revisión independiente evalúa cada resultado y una persona decide si se avanza. Ejecutar: la IA escribe el código siguiendo reglas, pero no decide cómo es la solución.

El principio que lo ordena todo: en SPAD, cumplir el flujo es más importante que la calidad aparente del resultado. La prioridad es, por este orden, el cumplimiento del proceso, la corrección técnica y la eficiencia.

Ejemplo: se pide a la IA un plan y entrega un buen plan seguido de quinientas líneas de código «de ejemplo». Aunque el código sea bueno, la respuesta se descarta entera y la fase se repite. ¿Por qué? Si se acepta «solo esta vez porque funciona», a la segunda ocasión el proceso deja de existir. Un «no» válido de una revisión vale más que un «sí» inválido.

La guía operativa (documento 01) fija doce reglas no negociables. Las tarjetas muestran tres de ellas. Otras son:

- Toda salida es auditable: documentos numerados y trazables.
- Toda decisión es explícita: no hay decisiones de diseño escondidas en el código.
- El código ejecuta, no decide.
- Revisiones independientes, también por una persona.
- Pruebas obligatorias, con cobertura mínima fijada antes de construir.
- Seguridad por diseño.
- Cada artefacto indica qué modelo y qué instrucciones lo produjeron.

Por qué importa a un directivo: es la separación de funciones de cualquier proceso serio (quien hace no se aprueba a sí mismo), aplicada también a las inteligencias artificiales. Y deja siempre a una persona como responsable.

Pregunta para el aula: ¿aceptaría en su compañía un informe financiero correcto que se saltó el control previo? ¿Por qué con el software sería distinto?

TÉRMINOS DE ESTA SLIDE

- **Principio de ejecución bloqueante**: Regla central de SPAD: ninguna fase empieza hasta que la anterior ha entregado su artefacto y ha recibido un veredicto favorable. Evita que la IA construya sobre un plan que nadie ha revisado. El enlace abre el principio en la guía operativa. https://seachad.github.io/AI_CONSULTING/SPAD/html/es/01_SPAD_Guia_operativa.html#el-principio-de-ejecucion-bloqueante
- **Veredictos de SPAD** (GO / NO-GO): Toda revisión de SPAD termina con uno de tres veredictos: GO (se continúa), GO con cambios (se continúa tras corregir lo indicado) o NO-GO (vuelve a la fase que corresponda). Un solo vocabulario en todas las revisiones evita interpretaciones. El enlace abre los veredictos en la guía operativa. https://seachad.github.io/AI_CONSULTING/SPAD/html/es/01_SPAD_Guia_operativa.html#veredictos
- **SPAD Lite**: Versión reducida de SPAD para cambios pequeños y de bajo riesgo, con solo tres artefactos. Mantiene la revisión y la validación humana, pero con menos pasos. Sirve para no cargar un cambio menor con todo el ciclo. El enlace abre la versión reducida en la guía operativa. https://seachad.github.io/AI_CONSULTING/SPAD/html/es/01_SPAD_Guia_operativa.html#version-reducida-spad-lite
- **Ciclo principal de SPAD** (fases 0–10): Las once fases de SPAD: contexto, plan, revisión del plan, guía de código, estrategia de pruebas, implementación, pruebas, revisión de pruebas, revisión del código, correcciones y gestión de versiones. Cada una deja un artefacto que se revisa. El enlace abre el ciclo en la guía operativa. https://seachad.github.io/AI_CONSULTING/SPAD/html/es/01_SPAD_Guia_operativa.html#ciclo-principal-fases-0-a-10
- **Política de validación**: Documento de SPAD que fija qué debe validar una persona, cuándo y con qué rigor, según el riesgo del cambio, y qué evidencia queda de esa validación. Es la garantía de que la IA no decide sola qué entra en el código. El enlace abre el documento. https://seachad.github.io/AI_CONSULTING/SPAD/html/es/03_SPAD_Politica_de_validacion.html

FUENTES Y ENLACES

- **SPAD 00 §3 · El principio que lo ordena todo** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/00_SPAD_Que_es_y_para_que_sirve.html#el-principio-que-lo-ordena-todo
- **SPAD 01 §7 · Reglas operativas no negociables** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/01_SPAD_Guia_operativa.html#reglas-operativas-no-negociables
- **SPAD 03 §2 · Principio fundamental de validación** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/03_SPAD_Politica_de_validacion.html#principio-fundamental
- **El principio que lo ordena todo** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/00_SPAD_Que_es_y_para_que_sirve.html#el-principio-que-lo-ordena-todo
- **Las doce reglas operativas** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/01_SPAD_Guia_operativa.html#reglas-operativas-no-negociables



Notas

El ciclo principal sirve para construir una funcionalidad nueva. Tiene once fases, de la 0 a la 10. La slide agrupa algunas para que quepan.

- Fase 0 · Contexto y objetivo. Una persona (el orquestador) escribe qué problema se resuelve, para quién, qué queda fuera y cómo se sabrá que el resultado es correcto. Sin criterios de éxito no hay fase 1.
- Fase 1 · PLAN. La IA planificadora diseña la arquitectura, registra cada decisión con sus alternativas y clasifica los riesgos. No escribe código.
- Fase 2 · Revisión del plan. Una IA revisora, distinta, evalúa el plan y emite un veredicto. No lo corrige.
- Fases 3 y 4 · Guía de código y estrategia de pruebas. Se traducen las decisiones en reglas verificables y se fija antes de construir qué se probará y con qué cobertura mínima. Son las fases que más se saltan y las que más valor protegen.
- Fases 5 y 6 · Implementación. La IA constructora escribe el código y las pruebas. Si el plan no prevé algo, no decide: lo señala y la persona decide.
- Fases 7 y 8 · Revisiones de pruebas y de código, también de las dependencias añadidas y sus licencias.
- Fase 9 · Correcciones mínimas, que vuelven a la revisión del código, no a reescribirlo todo.
- Fase 10 · Gestión de versiones: número de versión, cambios, plan de despliegue con reversión probada y qué vigilar. Es la entrega a operación.

No todo el trabajo es una funcionalidad nueva. SPAD tiene ciclos complementarios con las mismas reglas: legado (código existente sin documentar), depuración (incidentes), corrección urgente (acelera las fases, no las elimina) y seguridad.

Por qué importa a un directivo: un error de arquitectura detectado en la fase 2 cuesta una conversación; detectado en producción, un incidente y a veces la confianza de un cliente.

TÉRMINOS DE ESTA SLIDE

- **Ciclo principal de SPAD** (fases 0–10): Las once fases de SPAD: contexto, plan, revisión del plan, guía de código, estrategia de pruebas, implementación, pruebas, revisión de pruebas, revisión del código, correcciones y gestión de versiones. Cada una deja un artefacto que se revisa. El enlace abre el ciclo en la guía operativa. https://seachad.github.io/AI_CONSULTING/SPAD/html/es/01_SPAD_Guia_operativa.html#ciclo-principal-fases-0-a-10
- **Veredictos de SPAD** (GO / NO-GO): Toda revisión de SPAD termina con uno de tres veredictos: GO (se continúa), GO con cambios (se continúa tras corregir lo indicado) o NO-GO (vuelve a la fase que corresponda). Un solo vocabulario en todas las revisiones evita interpretaciones. El enlace abre los veredictos en la guía operativa. https://seachad.github.io/AI_CONSULTING/SPAD/html/es/01_SPAD_Guia_operativa.html#veredictos
- **Roles de SPAD**: Los papeles que intervienen en SPAD: el orquestador (una persona que dirige el ciclo y valida) y varias IA con funciones separadas: planificadora, revisora, constructora, correctora, analista y de diagnóstico. Quien construye no revisa lo que construye. El enlace abre el documento de presentación de SPAD, que describe cada rol. https://seachad.github.io/AI_CONSULTING/SPAD/html/es/00_SPAD_Que_es_y_para_que_sirve.html

FUENTES Y ENLACES

- **SPAD 00 §5 · El ciclo principal** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/00_SPAD_Que_es_y_para_que_sirve.html#el-ciclo-principal
- **SPAD 04 · Ciclos complementarios** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/04_SPAD_Ciclos_complementarios.html
- **Otros ciclos de trabajo** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/04_SPAD_Ciclos_complementarios.html



Notas

SPAD separa los roles lógicos de las personas y de las inteligencias artificiales que los ejercen.

- Orquestador (persona). Define el objetivo y el contexto, valida cada salida, aplica la política de validación y decide si se avanza. Tiene la última palabra. No puede delegar la validación en la IA que produjo la salida.
- IA planificadora. Diseña la arquitectura, la guía de código, la estrategia de pruebas y la versión. No escribe código.
- IA constructora. Implementa el código y las pruebas según el plan. No toma decisiones de diseño.
- IA correctora. Aplica correcciones mínimas y justificadas. No rediseña ni amplía funcionalidad.
- IA revisora. Revisa planes, pruebas y código y emite un veredicto. No corrige lo que revisa ni aprueba con problemas conocidos.
- Otros roles: IA analista (documenta código heredado), IA de diagnóstico (analiza incidentes) e IA revisora de seguridad.

La regla clave: una misma IA puede asumir varios roles a lo largo de un trabajo, pero nunca dos en la misma fase, y la IA que revisa un documento nunca es la que lo produjo.

Revisión humana del código. Dos inteligencias artificiales pueden coincidir en el mismo error. Por eso, además, una persona competente revisa el código antes de incorporarlo y deja registro. En código sensible (autenticación, pagos, datos personales, límites de agentes) hay una segunda revisión humana, por alguien que no orquestó la generación.

Un matiz de nombres: el rol que revisa se llama «IA revisora» y no «auditor», para no confundirlo con un auditor humano. Es una IA: su veredicto lo acepta siempre una persona.

Por qué importa a un directivo: es el control clásico de separación de funciones. Quien responde ante la organización de que el código es correcto es una persona, y esa responsabilidad no se delega en una

herramienta.

TÉRMINOS DE ESTA SLIDE

- **Roles de SPAD**: Los papeles que intervienen en SPAD: el orquestador (una persona que dirige el ciclo y valida) y varias IA con funciones separadas: planificadora, revisora, constructora, correctora, analista y de diagnóstico. Quien construye no revisa lo que construye. El enlace abre el documento de presentación de SPAD, que describe cada rol. https://seachad.github.io/AI_CONSULTING/SPAD/html/es/00_SPAD_Que_es_y_para_que_sirve.html
- **Veredictos de SPAD** (GO / NO-GO): Toda revisión de SPAD termina con uno de tres veredictos: GO (se continúa), GO con cambios (se continúa tras corregir lo indicado) o NO-GO (vuelve a la fase que corresponda). Un solo vocabulario en todas las revisiones evita interpretaciones. El enlace abre los veredictos en la guía operativa. https://seachad.github.io/AI_CONSULTING/SPAD/html/es/01_SPAD_Guia_operativa.html#veredictos

FUENTES Y ENLACES

- **SPAD 01 §3 · Roles** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/01_SPAD_Guia_operativa.html#roles
- **SPAD 01 §3.1 · Revisión humana del código** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/01_SPAD_Guia_operativa.html#3-1-revision-humana-del-codigo
- **SPAD 03 §3.5 · Autoaprobación** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/03_SPAD_Politica_de_validacion.html#3-5-autoaprobacion
- **Roles de SPAD** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/01_SPAD_Guia_operativa.html#roles
- **Revisión humana del código** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/01_SPAD_Guia_operativa.html#3-1-revision-humana-del-codigo



SPAD EN 10 SLIDES

Política de validación: veredictos y control humano

Todas las revisiones de la política de validación usan el mismo vocabulario: la IA propone el veredicto; una persona lo acepta o rechaza.

Situación	Qué significa	Qué ocurre
GO	El documento cumple.	Se pasa a la fase siguiente.
GO con cambios	Válido en lo esencial, con ajustes concretos.	Se ajusta y se revisa de nuevo.
NO-GO	No es aceptable: hay que rehacer.	Vuelve a la fase que corresponde. Es el proceso funcionando.
Respuesta que viola su fase	Por ejemplo, un plan con código o una revisión que corrige.	Se descarta entera y se repite la fase.
Autoaprobación	«Hay tres problemas, pero son menores: GO.»	Descarte inmediato: infracción crítica.
Revisión humana del código	Una persona competente revisa el código y deja registro.	Segunda revisión humana en código sensible.

Un NO-GO válido siempre es preferible a un GO inválido.

→ Veredictos → Política de validación

SPAD · Versión 0.x, en revisión · CC BY 4.0 · Fernando García Varela seachad.github.io/AI_CONSULTING/ 7 / 10

Notas

Las revisiones de SPAD (del plan, de las pruebas, del código y de seguridad) usan solo tres veredictos.

- GO: el documento cumple y se avanza.
- GO con cambios: es válido en lo esencial, pero necesita ajustes concretos y una nueva revisión.
- NO-GO: no es aceptable y hay que rehacer. Por ejemplo, un NO-GO en la revisión del plan devuelve el trabajo a la fase 0, porque el problema está mal planteado.

Un solo vocabulario permite comparar fases, medir el proceso y automatizar comprobaciones. Un veredicto sin justificación escrita está incompleto y se descarta.

La política de validación es una regla para personas, no un texto para la IA. La aplica el orquestador y define cuándo una respuesta es inválida aunque parezca buena. Hay cinco causas:

- Violación de fase: la respuesta hace algo que no toca en esa fase.
- Documento incompleto: por ejemplo, una revisión sin veredicto.
- Decisiones fuera del plan: por ejemplo, la implementación añade una caché que el plan no preveía.
- Implementación sin revisión previa: se descarta todo el código.
- Autoaprobación: la IA se aprueba a sí misma o minimiza problemas. Es la más grave.

Lo que no invalida: un NO-GO bien fundado, una vulnerabilidad detectada o una prueba que falla. Eso es el proceso funcionando.

Si una violación se repite, se refuerza la instrucción. Si persiste tras el máximo de intentos fijado (valor de partida: tres), se cambia de modelo o interviene una persona.

Por qué importa a un directivo: este es el punto donde la organización ejerce el control real sobre lo que se construye. Una revisión que nunca dice NO-GO durante meses es una señal de alarma: probablemente se ha convertido en un trámite.

Pregunta para el aula: ¿cómo premia su organización al revisor que dice «no»?

TÉRMINOS DE ESTA SLIDE

- **Política de validación**: Documento de SPAD que fija qué debe validar una persona, cuándo y con qué rigor, según el riesgo del cambio, y qué evidencia queda de esa validación. Es la garantía de que la IA no decide sola qué entra en el código. El enlace abre el documento. https://seachad.github.io/AI_CONSULTING/SPAD/html/es/03_SPAD_Politica_de_validacion.html
- **Veredictos de SPAD** (GO / NO-GO): Toda revisión de SPAD termina con uno de tres veredictos: GO (se continúa), GO con cambios (se continúa tras corregir lo indicado) o NO-GO (vuelve a la fase que corresponda). Un solo vocabulario en todas las revisiones evita interpretaciones. El enlace abre los veredictos en la guía operativa. https://seachad.github.io/AI_CONSULTING/SPAD/html/es/01_SPAD_Guia_operativa.html#veredictos
- **Revisión humana del código**: En SPAD, el código que genera la IA siempre lo revisa una persona antes de integrarlo, y hay una segunda revisión en código sensible, agentes con autonomía y producción crítica. La IA ayuda a construir, pero no se aprueba a sí misma. El enlace abre la regla en la guía operativa. https://seachad.github.io/AI_CONSULTING/SPAD/html/es/01_SPAD_Guia_operativa.html#3-1-revision-humana-del-codigo

FUENTES Y ENLACES

- **SPAD 03 §3 · Causas de invalidación** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/03_SPAD_Politica_de_validacion.html#causas-de-invalidacion
- **SPAD 03 §4 · Lo que no invalida una respuesta** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/03_SPAD_Politica_de_validacion.html#lo-que-no-invalida-una-respuesta
- **SPAD 03 §5 · Procedimiento ante una respuesta inválida** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/03_SPAD_Politica_de_validacion.html#procedimiento-ante-una-respuesta-invalida



Notas

Antes de cualquier fase se cargan dos contextos. Son las reglas con las que trabaja la IA.

- Contexto global: reglas comunes a todos los proyectos. Dónde se guardan los documentos, cobertura mínima de pruebas, versionado, seguridad base, modelos aprobados por rol y qué clases de información pueden enviarse a cada proveedor de IA. Ejemplo ilustrativo: «Ningún dato personal real en instrucciones a proveedores externos; datos de prueba sintéticos».
- Contexto de proyecto: lenguaje, arquitectura, reglas de negocio y obligaciones de cumplimiento. Ejemplo ilustrativo: «Los datos de clientes no salen de la región europea».

Si chocan, prevalece el de proyecto, pero solo con una excepción escrita, con su riesgo, evaluada por la IA revisora y aprobada por una persona. Ninguna IA aprueba una excepción.

El tema (TOPIC) es un identificador corto, por ejemplo payment_gateway, que agrupa en una carpeta todo el trabajo: plan, revisiones, código, pruebas, correcciones y versión. Un plan rechazado no se borra: se conserva con su revisión y la siguiente iteración toma el número siguiente. Así se ve qué se intentó y por qué se rechazó.

Cada documento lleva tres registros:

- Del modelo: qué herramienta lo produjo. Sin él, un resultado no se puede reproducir cuando cambia el proveedor.
- De validación humana: quién lo aceptó. Sin él, la trazabilidad solo prueba lo que hizo la IA.
- De infracciones: cada respuesta descartada, con fecha, fase, causa y modelo.

Por qué importa a un directivo: responden a las tres preguntas de cualquier auditoría del desarrollo. Además, la sección de datos y proveedores es la que más se olvida y la que más daño evita: sin ella, cada persona decide qué datos o secretos pega en una herramienta externa.

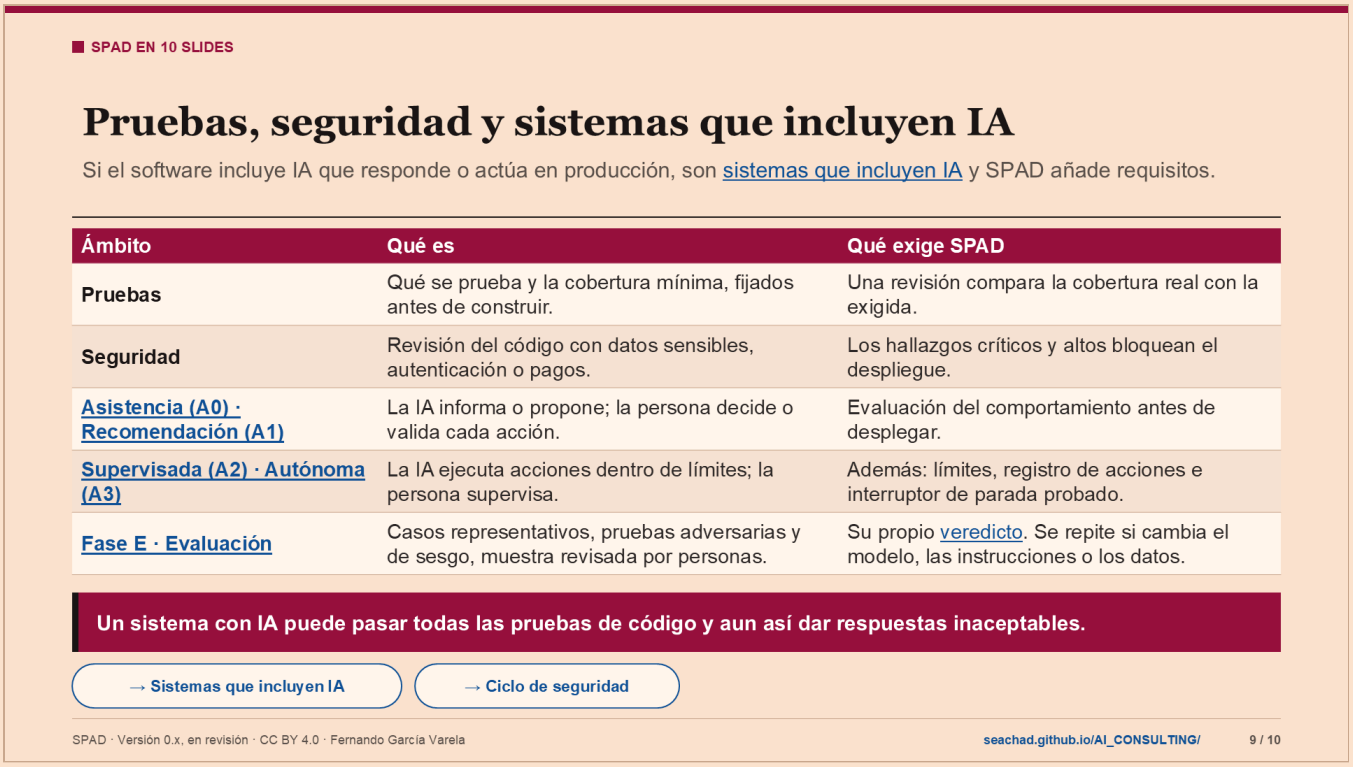
Pregunta para el aula: si mañana le piden explicar por qué un sistema crítico se construyó así, ¿dónde buscaría?

TÉRMINOS DE ESTA SLIDE

- **Contextos de SPAD**: Documentos de reglas que la IA debe respetar en cada fase: el contexto global (normas de la organización, datos, modelos aprobados) y el contexto de proyecto. Las excepciones las evalúa la IA revisora y las aprueba una persona. Sirven para que la IA no trabaje con supuestos propios. El enlace abre los contextos. https://seachad.github.io/AI_CONSULTING/SPAD/html/es/02_SPAD_Contextos_temas_y_registro.html#contextos
- **Temas de SPAD**: Un tema es la unidad de trabajo de SPAD: una funcionalidad o un cambio que recorre el ciclo completo con su propia carpeta de artefactos. Permite seguir de principio a fin qué se pidió, qué se planificó, qué se construyó y quién lo validó. El enlace abre su definición. https://seachad.github.io/AI_CONSULTING/SPAD/html/es/02_SPAD_Contextos_temas_y_registro.html#temas
- **Registros por artefacto**: Cada artefacto de SPAD lleva tres registros: del modelo (proveedor, modelo, versión, instrucción y contextos usados), de validación humana (quién lo revisó y cuándo) y de infracciones (reglas que no se cumplieron). Hacen el proceso auditable. El enlace abre los registros. https://seachad.github.io/AI_CONSULTING/SPAD/html/es/02_SPAD_Contextos_temas_y_registro.html#4-1-registro-del-modelo
- **Métricas del proceso**: Indicadores que mide un equipo que usa SPAD, como cuántas revisiones terminan en NO-GO, cuántos defectos llegan a producción o cuánto tarda un tema. SPAD no promete cifras: propone medirlas y fijar objetivos propios. El enlace abre las métricas. https://seachad.github.io/AI_CONSULTING/SPAD/html/es/08_SPAD_Autoevaluacion_y_metricas.html#metricas-del-proceso

FUENTES Y ENLACES

- **SPAD 02 §4 · Registros por artefacto** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/02_SPAD_Contextos_temas_y_registro.html#registros-por-artefacto
- **Contextos, temas y registro** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/02_SPAD_Contextos_temas_y_registro.html
- **Lista antes de empezar** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/02_SPAD_Contextos_temas_y_registro.html#lista-de-comprobacion-antes-de-empezar



SPAD EN 10 SLIDES

Pruebas, seguridad y sistemas que incluyen IA

Si el software incluye IA que responde o actúa en producción, son sistemas que incluyen IA y SPAD añade requisitos.

Ámbito	Qué es	Qué exige SPAD
Pruebas	Qué se prueba y la cobertura mínima, fijados antes de construir.	Una revisión compara la cobertura real con la exigida.
Seguridad	Revisión del código con datos sensibles, autenticación o pagos.	Los hallazgos críticos y altos bloquean el despliegue.
Asistencia (A0) · Recomendación (A1)	La IA informa o propone; la persona decide o valida cada acción.	Evaluación del comportamiento antes de desplegar.
Supervisada (A2) · Autónoma (A3)	La IA ejecuta acciones dentro de límites; la persona supervisa.	Además: límites, registro de acciones e interruptor de parada probado.
Fase E · Evaluación	Casos representativos, pruebas adversarias y de sesgo, muestra revisada por personas.	Su propio veredicto. Se repite si cambia el modelo, las instrucciones o los datos.

Un sistema con IA puede pasar todas las pruebas de código y aun así dar respuestas inaceptables.

→ Sistemas que incluyen IA → Ciclo de seguridad

SPAD · Versión 0.x, en revisión · CC BY 4.0 · Fernando García Varela seachad.github.io/AI_CONSULTING/ 9 / 10

Notas

Pruebas. En SPAD, la estrategia de pruebas se escribe antes de construir (fase 4): qué casos, qué tipos de prueba y qué cobertura mínima. Así las pruebas no se escriben al final para confirmar lo que ya existe. Después, una revisión independiente (fase 7) compara la cobertura real con la exigida. Las pruebas que solo repiten el código sin verificar el comportamiento esperado no cuentan.

Seguridad. El código que maneja datos sensibles, autenticación, autorización o pagos pasa una revisión de seguridad, con guías reconocidas como el OWASP Top 10 de referencia. Los hallazgos se clasifican por severidad: los críticos y los altos bloquean el despliegue; el riesgo residual lo acepta siempre una persona, nunca una IA. También se revisan las dependencias que propone la IA (existencia, procedencia, vulnerabilidades y licencia).

Sistemas que incluyen IA. No es lo mismo usar IA para escribir código que construir un sistema cuyo comportamiento depende de un modelo en producción. En el segundo caso pueden aparecer respuestas incorrectas con mucha seguridad, inyección de instrucciones, fuga de datos, acciones no autorizadas, sesgo o deriva. Y el comportamiento puede cambiar sin tocar el código: basta con que el proveedor actualice el modelo.

Por eso SPAD clasifica cada componente con una escala de autonomía de cuatro niveles, de A0 (asistencia) a A3 (actuación autónoma). Ejemplos ilustrativos: A0, un resumen de un expediente para un analista; A3, un ajuste continuo de precios dentro de una banda aprobada. Y añade la fase E, evaluación del comportamiento, entre la revisión de pruebas y la del código.

Por qué importa a un directivo: sin evaluación del comportamiento, la única prueba de que un asistente responde bien es la demostración que alguien hizo con las preguntas que se le ocurrieron.

Pregunta para el aula: ¿qué hace su compañía cuando el proveedor actualiza el modelo de un asistente que ya usan sus clientes?

TÉRMINOS DE ESTA SLIDE

- **Sistemas que incluyen IA**: Qué añade SPAD cuando lo que se construye es un sistema que usa IA: niveles de autonomía, una fase de evaluación del comportamiento con conjuntos de prueba, pruebas adversarias y de sesgo, plan de agentes y control de cambios que no tocan el código (como cambiar el modelo). El enlace abre el documento. https://seachad.github.io/AI_CONSULTING/SPAD/html/es/05_SPAD_Sistemas_que_incluyen_IA.html
- **Veredictos de SPAD** (GO / NO-GO): Toda revisión de SPAD termina con uno de tres veredictos: GO (se continúa), GO con cambios (se continúa tras corregir lo indicado) o NO-GO (vuelve a la fase que corresponda). Un solo vocabulario en todas las revisiones evita interpretaciones. El enlace abre los veredictos en la guía operativa. https://seachad.github.io/AI_CONSULTING/SPAD/html/es/01_SPAD_Guia_operativa.html#veredictos

FUENTES Y ENLACES

- **SPAD 05 · Sistemas que incluyen IA** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/05_SPAD_Sistemas_que_incluyen_IA.html#dos-objetos-distintos
- **SPAD 05 §4 · Fase E · Evaluación del comportamiento** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/05_SPAD_Sistemas_que_incluyen_IA.html#fase-e-evaluacion-del-comportamiento
- **SPAD 04 §5 · Ciclo de seguridad** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/04_SPAD_Ciclos_complementarios.html#ciclo-de-seguridad
- **OWASP Top 10 para aplicaciones con modelos de lenguaje (2025)** https://genai.owasp.org/resource/owasp-top-10-for-llm-applications-2025/
- **Ciclo de seguridad** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/04_SPAD_Ciclos_complementarios.html#ciclo-de-seguridad



Notas

Medir. SPAD no afirma cifras de mejora. Cada organización mide si el método funciona frente a su propia situación de partida, con indicadores que salen de los registros: tasa de invalidación por fase y por modelo, tasa de NO-GO, iteraciones hasta GO, tiempo por fase, cobertura real frente a exigida, correcciones sin hallazgo y defectos escapados a producción, que es la cifra que de verdad importa. Una tasa de NO-GO cercana a cero durante meses indica que la revisión se ha vuelto un trámite.

Versión reducida (SPAD Lite). Para cambios acotados, sin datos personales ni dinero, con un equipo pequeño. Agrupa las fases en tres documentos. No recorta la separación entre quien construye y quien revisa, la validación humana, la política de validación ni el registro del tema.

Autoevaluación. Una persona independiente de quienes construyeron comprueba seis condiciones: decisiones antes del código, un documento por fase, revisiones independientes, ninguna fase saltada, correcciones mínimas y ninguna aprobación de una IA. No es una certificación: no la emite ni la respalda su autor ni ningún tercero.

Para qué le sirve a su compañía (posición del autor, sin cifras). Primer uso: construir su propio software con IA con método. Segundo uso, como cliente: una compañía que sabe cómo se construye software con IA puede exigir a sus proveedores e integradores que la ganancia de productividad llegue a ella, en menos coste o en más alcance por el mismo precio. Puede pedirles lo que SPAD ya define: el artefacto de cada fase, el registro del modelo y de la validación humana, los veredictos y las métricas del proceso. No fija un porcentaje: permite negociar con datos.

Un marco aparte. SPAD no es gobierno de la IA de la compañía. Quien quiera gobernar qué se construye, con qué riesgo y cómo se mide su valor tiene un marco aparte, SEVEN-G; el único puente es su documento 53, y los veredictos de SPAD no sustituyen a sus puertas de decisión.

Cómo empezar: elija un trabajo acotado, use la versión reducida y siga la lista de comprobación del documento 02.

Por qué importa a un directivo: un consejo no necesita saber cómo se revisa un plan, pero sí que existe un método que lo hace y deja evidencias.

TÉRMINOS DE ESTA SLIDE

- **Métricas del proceso**: Indicadores que mide un equipo que usa SPAD, como cuántas revisiones terminan en NO-GO, cuántos defectos llegan a producción o cuánto tarda un tema. SPAD no promete cifras: propone medirlas y fijar objetivos propios. El enlace abre las métricas. https://seachad.github.io/AI_CONSULTING/SPAD/html/es/08_SPAD_Autoevaluacion_y_metricas.html#metricas-del-proceso
- **SPAD Lite**: Versión reducida de SPAD para cambios pequeños y de bajo riesgo, con solo tres artefactos. Mantiene la revisión y la validación humana, pero con menos pasos. Sirve para no cargar un cambio menor con todo el ciclo. El enlace abre la versión reducida en la guía operativa. https://seachad.github.io/AI_CONSULTING/SPAD/html/es/01_SPAD_Guia_operativa.html#version-reducida-spad-lite
- **Autoevaluación de conformidad**: Revisión con la que un equipo comprueba si aplica SPAD como dice: qué fases sigue, qué registros guarda y qué falla. Vale doce meses y no la otorga nadie desde fuera. Sirve para mejorar el proceso con datos y no con impresiones. El enlace abre la autoevaluación. https://seachad.github.io/AI_CONSULTING/SPAD/html/es/08_SPAD_Autoevaluacion_y_metricas.html#que-es-la-autoevaluacion
- **Ciclo principal de SPAD** (fases 0–10): Las once fases de SPAD: contexto, plan, revisión del plan, guía de código, estrategia de pruebas, implementación, pruebas, revisión de pruebas, revisión del código, correcciones y gestión de versiones. Cada una deja un artefacto que se revisa. El enlace abre el ciclo en la guía operativa. https://seachad.github.io/AI_CONSULTING/SPAD/html/es/01_SPAD_Guia_operativa.html#ciclo-principal-fases-0-a-10
- **Contextos de SPAD**: Documentos de reglas que la IA debe respetar en cada fase: el contexto global (normas de la organización, datos, modelos aprobados) y el contexto de proyecto. Las excepciones las evalúa la IA revisora y las aprueba una persona. Sirven para que la IA no trabaje con supuestos propios. El enlace abre los contextos. https://seachad.github.io/AI_CONSULTING/SPAD/html/es/02_SPAD_Contextos_temas_y_registro.html#contextos
- **Temas de SPAD**: Un tema es la unidad de trabajo de SPAD: una funcionalidad o un cambio que recorre el ciclo completo con su propia carpeta de artefactos. Permite seguir de principio a fin qué se pidió, qué se planificó, qué se construyó y quién lo validó. El enlace abre su definición. https://seachad.github.io/AI_CONSULTING/SPAD/html/es/02_SPAD_Contextos_temas_y_registro.html#temas

FUENTES Y ENLACES

- **SPAD 08 §2 · Condiciones de conformidad** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/08_SPAD_Autoevaluacion_y_metricas.html#condiciones-de-conformidad
- **SPAD 00 · Para qué le sirve a una compañía** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/00_SPAD_Que_es_y_para_que_sirve.html#para-que-le-sirve-a-una-compania
- **SEVEN-G 53 · Construcción de soluciones con IA (único puente)** https://seachad.github.io/AI_CONSULTING/SEVEN-G/html/es/53_SEVEN-G_Construccion_de_soluciones_con_IA.html
- **SPAD 02 §5 · Lista de comprobación antes de empezar** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/02_SPAD_Contextos_temas_y_registro.html#lista-de-comprobacion-antes-de-empezar
- **Autoevaluación y métricas** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/08_SPAD_Autoevaluacion_y_metricas.html
- **Para qué le sirve a una compañía** https://seachad.github.io/AI_CONSULTING/SPAD/html/es/00_SPAD_Que_es_y_para_que_sirve.html#para-que-le-sirve-a-una-compania

AVISO LEGAL

Aviso legal: ayuda metodológica genérica y gratuita, «tal cual» y con fines informativos; no es asesoramiento jurídico ni está revisada jurídicamente para ningún caso. Cada organización la adapta y es responsable de su revisión legal y de su cumplimiento regulatorio; puede no recoger cambios normativos recientes. En la máxima medida permitida por la ley, el autor no asume responsabilidad alguna por los efectos de su aplicación. No otorga certificación de ningún tipo. Detalle en el documento 93.

CC BY 4.0 (contenidos) · MIT (código) · Fernando García Varela · seachad.github.io/AI_CONSULTING/