/thoughtworks

Diseño. Ingeniería. IA.

Radar Tecnológico

Una guía de opinión sobre el entorno tecnológico actual

Volumen 33
Nov. 2025

Sobre el Radar

Los Thoughtworkers son personas a las que les apasiona la tecnología. La construimos, la investigamos, la probamos, abogamos por el código abierto, escribimos sobre ella y constantemente tratamos de mejorarla para todas las personas. Nuestra misión es defender la excelencia del software y evolucionar la TI. Creamos y compartimos el Radar Tecnológico de Thoughtworks en apoyo de esa misión. El Technology Advisory Board de Thoughtworks, un grupo de líderes tecnológicos de alto nivel de Thoughtworks, crea el Radar. Se reúnen periódicamente para debatir la estrategia tecnológica global de Thoughtworks y las tendencias tecnológicas que tienen un impacto significativo en nuestra industria.

El Radar recoge el resultado de los debates del Technology Advisory Board en un formato que proporciona valor a una amplia gama de partes interesadas, desde las personas desarrolladoras hasta CTOs. El contenido pretende ser un resumen conciso.

Te animamos a explorar estas tecnologías. El Radar es de naturaleza gráfica y agrupa los elementos en técnicas, herramientas, plataformas, lenguajes y frameworks. Cuando los elementos del Radar podían aparecer en varios cuadrantes, elegimos el que nos pareció más apropiado. Además, agrupamos estos elementos en cuatro anillos para reflejar nuestra posición actual al respecto

Para más información sobre el Radar,consulta thoughtworks.com/es/radar/faq.

Un vistazo al Radar

El Radar se dedica a rastrear cosas interesantes, a las que nos referimos como blips. Organizamos los blips en el Radar utilizando dos elementos de categorización: cuadrantes y anillos. Los cuadrantes representan los diferentes tipos de blips. Los anillos indican nuestra recomendación para utilizar esa tecnología.

Un blip es una tecnología o técnica que desempeña un papel en el desarrollo de software. Los blips son cosas que están en "movimiento", es decir, que su posición en el Radar cambia a menudo, lo que suele indicar nuestra creciente confianza en recomendarlos a medida que avanzan por los anillos.

Adoptar: Estamos convencidas de que la industria debería Adoptarar estos ítems Nosotras los utilizamos cuando es apropiado en nuestros proyectos.

Probar: Vale la pena probarlos. Es importante entender cómo desarrollar estas capacidades. Las empresas deberían probar esta tecnología en proyectos en que se puede manejar el riesgo.

Evaluar: Vale la pena explorar con el objetivo de comprender cómo afectará a su empresa.

Resistir: Proceder con precaución.

Nuevo **Deplazado adentro/afuera** **Ningún cambio**

Nuestro Radar está orientado al futuro. Para dar paso a nuevos artículos, desvanecemos los queno se han movido recientemente, lo cual no es un reflejo de su valor, sino de nuestro limitado espacio en el Radar.

Contribuyentes

El Technology Advisory Board (TAB) es un grupo de 22 personas tecnólogas senior de Thoughtworks. El TAB se reúne dos veces al año en persona y virtualmente cada dos semanas. Su función principal es ser un grupo de asesoramiento para la CTO de Thoughtworks Rachel Laycock.

El TAB actúa como un organismo amplio que puede examinar los temas que afectan a la tecnología y a las personas tecnólogas en Thoughtworks. Esta edición del Radar Tecnológico de Thoughtworks es en base a la reunión virtual que TAB realizó remotamente en Septiembre del 2025.


Rachel Laycock (CTO)


Martin Fowler (Chief Scientist)

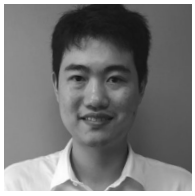

Alessio Ferri


Bharani Subramaniam


Birgitta Böckeler


Bryan Oliver


Camilla Falconi Crispim


Chris Chakrit Riddhagni


Effy Elden


James Lewis


Kief Morris


Ken Mugrage


Maya Ormaza


Nati Rivera


Neal Ford


Ni Wang


Nimisha Asthagiri


Pawan Shah


Selvakumar Natesan


Shangqi Liu


Vanya Seth


Will Amaral

Créditos de producción

Personas editoras

- **Willian Amaral**
  Product Owner
- **Nati Rivera**
  Product Owner
- **Preeti Mishra**
  Project and Campaign Manager
- **Richard Gall**
  Content Editor
- **Michael Koch**
  Copy Editor
- **Gareth Morgan**
  Head of Content and Thought Leadership
- **Carlos Oquendo**
  Principal Developer
- **Cecilia Geraldo**
  Lead Developer
- **Eduard Maura i Puig**
  Senior Consultant
- **Esteban Moreno**
  Lead Developer
- **Fernando Tamayo**
  Principal Developer
- **Jorge Arimany**
  Lead Developer
- **Juan Romero**
  Consultant Developer
- **Sergio Gutiérrez Santos**
  Lead Developer
- **Veronica Rodriguez**
  Principal Developer

Experiencia digital y web

- **Rashmi Naganur**
  Business Analyst
- **Brigitte Britten-Kelly**
  Digital Content Strategist
- **Vandita Kamboj**
  UX Designer
- **Lohith Amruthappa**
  Analytics Specialist
- **Neeti Thakur**
  Marketing Automation Specialist

Diseño y multimedia

- **Leticia Nunes**
  Lead Designer
- **Sruba Deb**
  Visual Designer
- **Ryan Cambage**
  Multimedia Specialist
- **Anish Thomas**
  Multimedia Designer

Marketing

- **Shalini Jagadish**
  Internal Communications Specialist
- **Hiral Shah**
  Social Media Specialist
- **Abhishek Kasegaonkar**
  Social Media Specialist
- **Michelle Surendran**
  Public Relations Specialist
- **Anushree Tapuriah**
  Campaigns and Advertisement Specialist
- **Prakhar Nigam**
  Campaigns and Advertisement Specialist
- **Daniel Negrete Banda**
  Marketing Lead - versión en español

Temas

La orquestación de infraestructura llega a la IA

Las cargas de trabajo de IA están impulsando a las organizaciones a orquestar grandes flotas de GPU tanto para entrenamiento como para inferencia. Los equipos trabajan cada vez más con modelos cuyo tamaño supera la capacidad de un solo acelerador (incluso con 80 GB de HBM), lo que los lleva hacia el entrenamiento distribuido y la inferencia multi-GPU. Como resultado, los equipos de plataforma están construyendo pipelines complejos de múltiples etapas y ajustando continuamente el rendimiento para optimizar el throughput y la latencia. Las discusiones en este espacio incluyeron el uso de Nvidia DCGM Exporter para la telemetría de flotas y topology-aware scheduling tpara ubicar los trabajos donde el ancho de banda entre interconexiones es mayor.

Antes de este auge en la demanda de GPU, Kubernetes ya se había consolidado como el orquestador de contenedores por defecto, y sigue siendo una base sólida para gestionar cargas de trabajo de IA a gran escala, incluso mientras exploramos alternativas como micro y Uncloud. Estamos siguiendo de cerca los patrones emergentes de planificación consciente de GPU, como la gestión de colas y cuotas mediante Kueue, combinada con la asignación según topología y gang scheduling, para ubicar trabajos multi-GPU en enlaces GPU-a-GPU rápidos (por ejemplo, NVLink/NVSwitch) y dentro de “islas” contiguas del centro de datos (como racks o pods con RDMA). Las mejoras recientes en las API de Kubernetes con reconocimiento multi-GPU y NUMA fortalecen aún más estas capacidades, mejorando el ancho de banda entre dispositivos, reduciendo la latencia de cola y aumentando la utilización efectiva.

Esperamos una rápida innovación en infraestructura de IA a medida que los equipos de plataforma buscan dar soporte al creciente volumen de flujos de trabajo de codificación asistida por IA y el auge de los agentes impulsados por MCP. En nuestra opinión, la orquestación consciente de GPU se está convirtiendo en un estándar básico: la topología ahora es una preocupación principal en la planificación.

El auge de los agentes impulsado por MCP

El auge simultáneo de MCP y los agentes, junto con el ecosistema de protocolos en expansión y herramientas construidas a su alrededor domina esta edición del Radar. Prácticamente todos los grandes proveedores están incorporando compatibilidad con MCP en sus herramientas lo cual tiene sentido: en muchos aspectos, MCP se ha convertido en el protocolo de integración definitivo para potenciar agentes y permitirles trabajar de manera eficiente y semi autónoma. Estas capacidades son fundamentales para hacer que los flujos de trabajo agéntico sean realmente productivos.

Observamos una innovación continua en los flujos de trabajo agéntico donde la ingeniería de contexto ha demostrado ser fundamental para optimizar tanto el comportamiento como el consumo

de recursos. Nuevos protocolos como A2A y AG-UI están reduciendo la cantidad de configuración necesaria para construir y escalar aplicaciones multiagente orientadas al usuario. En el ámbito del desarrollo de software analizamos distintas formas de proporcionar contexto a los agentes de codificación desde archivos AGENTS.md hasta patrones como anclar agentes de codificación a una aplicación de referencia. Como es habitual en el ecosistema de IA cada edición del Tech Radar trae una nueva ola de innovación: la anterior fue RAG; esta vez, son los flujos de trabajo agéntico y la creciente constelación de herramientas, técnicas y plataformas que los sustentan junto con algunos antipatrones de IA emergentes que vale la pena observar.

Flujos de trabajo de codificación con IA

IEs evidente que la inteligencia artificial está transformando la forma en que construimos y mantenemos software, y sigue dominando nuestras conversaciones más recientes. A medida que la IA se integra estratégicamente a lo largo de toda la cadena de valor del software, desde el uso de IA para comprender bases de código heredadas hasta la ingeniería progresiva con GenAI estamos aprendiendo a suministrar mejor el conocimiento que necesitan los agentes de codificación. Los equipos están experimentando con nuevas prácticas, cómo definir prompts personalizados mediante archivos AGENTS.md e integrarse con servidores MCP como Context7 tpara obtener documentación de dependencias actualizada.

También existe una comprensión cada vez mayor de que la IA debe potenciar al equipo completo, no solo a las personas desarrolladoras. Están surgiendo técnicas como los prompts compartidos seleccionados y los comandos personalizados diseñadas para garantizar una difusión equitativa del conocimiento. El panorama de herramientas es vibrante: las personas diseñadoras exploran UX Pilot y AI Design Reviewer, mientras las desarrolladoras prototipan rápidamente con v0 y Bolt para la creación autónoma de prototipos de interfaz de usuario.

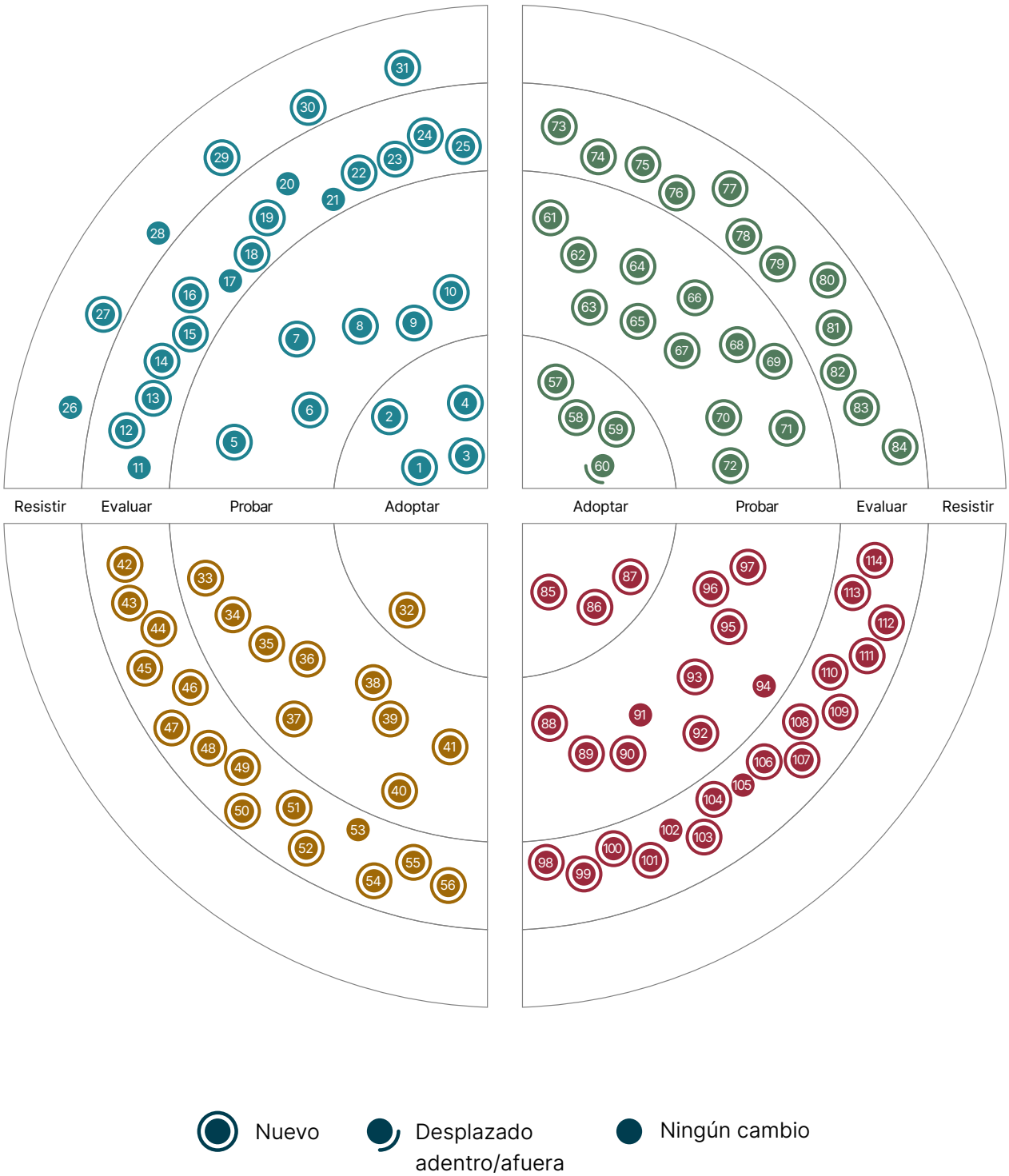
También seguimos debatiendo el desarrollo guiado por especificaciones: su alcance, nivel de granularidad y su potencial para servir como una única fuente de verdad en la entrega incremental. Sin embargo, en medio de este entusiasmo, la complacencia con el código generado por IA sigue siendo una preocupación compartida, recordándonos que, aunque la IA puede acelerar la ingeniería, el juicio humano sigue siendo indispensable.

Antipatrones emergentes de IA

La adopción acelerada de la IA en distintas industrias ha revelado tanto prácticas efectivas como antipatrones emergentes. Aunque vemos una clara utilidad en conceptos como la creación autónoma de prototipos de interfaz de usuario con GenAI, también reconocemos su potencial para llevar a las organizaciones hacia el antipatrón de TI en la sombra acelerado por IA. De manera similar, a medida que el Model Context Protocol (MCP) gana terreno, muchos equipos están cayendo en el antipatrón de conversión ingenua de API a MCP.

También hemos comprobado que la eficacia de las soluciones de texto-a-SQL no ha alcanzado las expectativas iniciales, y que la complacencia con código generado por IA sigue siendo un tema relevante. Incluso dentro de prácticas emergentes como el desarrollo guiado por especificaciones, hemos notado el riesgo de volver a antipatrones tradicionales de la ingeniería de software — en particular, una tendencia hacia la especificación excesiva al inicio y los lanzamientos de tipo “big bang”. Dado que la GenAI avanza a un ritmo y escala sin precedentes, esperamos que nuevos antipatrones sigan surgiendo rápidamente. Los equipos deben mantenerse atentos ante patrones que parecen efectivos al principio, pero que con el tiempo se degradan y reducen la retroalimentación, disminuyen la adaptabilidad o diluyen la responsabilidad.

El Radar



El Radar

Técnicas

Adoptar

1. Cumplimiento continuo
2. Prompts compartidos seleccionados para equipos de software
3. Hooks de pre-commit
4. Uso de GenAI para comprender bases de código heredadas

Probar

5. AGENTS.md
6. IA para migraciones de código
7. Delta Lake liquid clustering
8. Autoservicio de prototipado UI con GenAI
9. Salida estructurada de LLMs
10. TCR (Test && Commit || Revert)

Evaluar

11. Pruebas de IU impulsadas por IA
12. Anclar los agentes de codificación a una aplicación de referencia
13. Ingeniería de contexto
14. GenAI para ingeniería progresiva
15. GraphQL como patrón de acceso a datos para LLMs
16. Flujos de conocimiento sobre reservas de conocimiento
17. LLM como juez
18. La recuperación de información en el dispositivo
19. SAIF
20. Service mesh sin sidecar
21. Small language models
22. Spec-driven development
23. Equipo de agentes de codificación
24. Programación consciente de la topología
25. Análisis de flujo tóxico para IA

Resistir

26. TI en la sombra acelerado por IA
27. Capacity-driven development
28. Complacencia con el código generado por IA
29. Conversión ingenua de API a MCP
30. Equipos de ingeniería de datos separados
31. Text to SQL

Plataformas

Adoptar

32. Arm en la nube

Probar

33. Apache Paimon
34. DataDog LLM Observability
35. Delta Sharing
36. Dovetail
37. Langdock
38. LangSmith
39. Model Context Protocol (MCP)
40. n8n
41. OpenThread

Evaluar

42. AG-UI Protocol
43. Agent-to-Agent (A2A) Protocol
44. Amazon S3 Vectors
45. Ardoq
46. CloudNativePG
47. Coder
48. Graft
49. groundcover
50. Karmada
51. OpenFeature
52. Oxide
53. Restate
54. SkyPilot
55. StarRocks
56. Uncloud

Resistir

—

El Radar

Herramientas

Adoptar

57. ClickHouse
58. NeMo Guardrails
59. pnpm
60. Pydantic

Probar

61. AI Design Reviewer
62. Barman
63. Claude Code
64. Cleanlab
65. Context7
66. Data Contract CLI
67. Databricks Assistant
68. Hoppscotch
69. NVIDIA DCGM Exporter
70. RelationalAI
71. UX Pilot
72. v0

Evaluar

73. Augment Code
74. Azure AI Document Intelligence
75. Docling
76. E2B
77. Editor Helix
78. Kueue
79. MCPScan.ai
80. oRPC
81. Power User para dbt
82. Serena
83. SweetPad
84. Tape/Z (Tools for Assembly Program Exploration for Z/OS)

Resistir

—

Lenguajes y Frameworks

Adoptar

85. Fastify
86. LangGraph
87. vLLM

Probar

88. Crossplane
89. DeepEval
90. FastMCP
91. LiteLLM
92. MLForecast
93. Nuxt
94. Phoenix
95. Presidio
96. Pydantic AI
97. Tauri

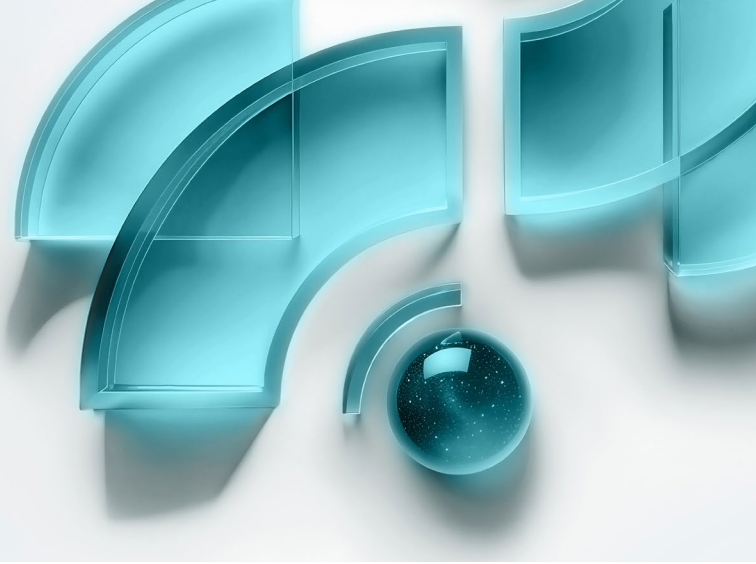
Evaluar

98. Agent Development Kit (ADK)
99. Agno
100. assistant-ui
101. AutoRound
102. Browser Use
103. DeepSpeed
104. Drizzle
105. Criptografía poscuántica en Java
106. kagent
107. LangExtract
108. Langflow
109. LMCache
110. Mem0
111. Open Security Control Assessment Language (OSCAL)
112. OpenInference
113. Valibot
114. Vercel AI SDK

Resistir

—

Técnicas



Adoptar

1. Cumplimiento continuo
2. Prompts compartidos seleccionados para equipos de software
3. Hooks de pre-commit
4. Uso de GenAI para comprender bases de código heredadas

Probar

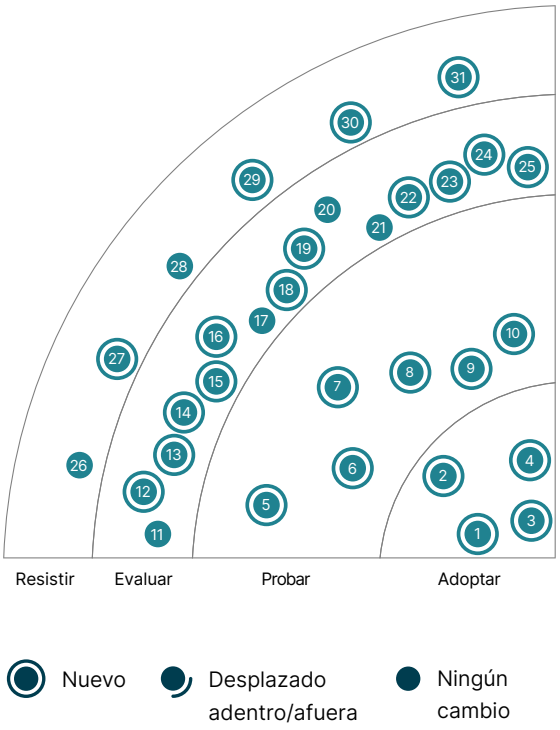
5. AGENTS.md
6. IA para migraciones de código
7. Delta Lake liquid clustering
8. Autoservicio de prototipado UI con GenAI
9. Salida estructurada de LLMs
10. TCR (Test && Commit || Revert)

Evaluar

11. Pruebas de IU impulsadas por IA
12. Anclar los agentes de codificación a una aplicación de referencia
13. Ingeniería de contexto
14. GenAI para ingeniería progresiva
15. GraphQL como patrón de acceso a datos para LLMs
16. Flujos de conocimiento sobre reservas de conocimiento
17. LLM como juez
18. La recuperación de información en el dispositivo
19. SAIF
20. Service mesh sin sidecar
21. Small language models
22. Spec-driven development
23. Equipo de agentes de codificación
24. Programación consciente de la topología
25. Análisis de flujo tóxico para IA

Resistir

26. TI en la sombra acelerado por IA
27. Capacity-driven development
28. Complacencia con el código generado por IA
29. Conversión ingenua de API a MCP
30. Equipos de ingeniería de datos separados
31. Text to SQL



Nuevo

Desplazado adentro/afuera

Ningún cambio

1. Cumplimiento continuo

Adoptar

Cumplimiento continuo es la práctica de garantizar que los procesos y tecnologías de desarrollo de software cumplan de manera constante con los estándares regulatorios y de seguridad mediante la automatización. Las verificaciones manuales de cumplimiento pueden ralentizar el desarrollo e introducir errores humanos, mientras que las auditorías y verificaciones automatizadas ofrecen retroalimentación más rápida, evidencia más clara y reportes simplificados. Al integrar herramientas de política como código como Open Policy Agent y generar SBOMs dentro de los pipelines de CD alineados con las guías de SLSA los equipos pueden detectar y resolver problemas de cumplimiento de manera temprana. La codificación de reglas y buenas prácticas permite aplicar los estándares de forma consistente entre equipos sin generar cuellos de botella. OSCAL también muestra un gran potencial como marco para automatizar el cumplimiento a gran escala. Las prácticas y herramientas para el cumplimiento continuo ya han madurado lo suficiente como para considerarse el estándar recomendado, por lo que hemos movido nuestra recomendación a Adoptar. El uso creciente de la IA en la programación y el riesgo asociado de complacencia con el código generado por IA hacen que incorporar el cumplimiento dentro del proceso de desarrollo sea más importante que nunca.

2. Prompts compartidos seleccionados para equipos de software

Adoptar

Para los equipos que utilizan activamente la IA en la entrega de software, el siguiente paso es ir más allá del prompting individual y avanzar hacia prompts compartidos seleccionados para equipos de software. Esta práctica ayuda a aplicar la IA de manera efectiva en todas las tareas de entrega (no solo en la codificación) mediante el uso compartido de prompts comprobados y de alta calidad. La forma más sencilla de implementarla es incorporando archivos de prompts, como AGENTS.md, directamente en el repositorio del proyecto. La mayoría de las herramientas de codificación con IA, incluidas Cursor, Windsurf y Claude Code permiten compartir prompts a través de comandos personalizados de barra (/) o flujos de trabajo. Para las tareas que no implican código, pueden configurarse bibliotecas de prompts a nivel organizacional listas para usar. Este enfoque sistemático permite la mejora continua: en cuanto un prompt se perfecciona, todo el equipo se beneficia, garantizando un acceso constante a los mejores prompts para interactuar con la IA.

3. Hooks de pre-commit

Adoptar

Los Git hooks existen desde hace mucho tiempo, pero consideramos que siguen siendo poco utilizados. El auge de la programación asistida por IA y agéntica ha incrementado el riesgo de hacer commit accidentalmente de secretos o de código problemático. Si bien existen muchos mecanismos de validación de código, como la integración continua, los hooks de pre-commit son una medida de protección simple y efectiva que más equipos deberían adoptar. Sin embargo, sobrecargar los hooks con verificaciones lentas puede desincentivar su uso, por lo que es mejor mantenerlos mínimos y enfocados en los riesgos que pueden detectarse de manera más eficaz en esta etapa del flujo de trabajo, como el escaneo de secretos.

4. Uso de GenAI para comprender bases de código heredadas

Adoptar

En los últimos meses, hemos visto evidencia clara de que usar GenAI para comprender bases de código heredadas puede acelerar significativamente el entendimiento de sistemas grandes y complejos. Herramientas como Cursor, Claude Code, Copilot, Windsurf, Aider, Cody, Swimm, Unblocked y PocketFlow-Tutorial-Codebase-Knowledge ayudan a los desarrolladores a identificar

reglas de negocio, resumir lógica e identificar dependencias. Usadas junto con frameworks abiertos y prompting directo a LLMs, reducen drásticamente el tiempo necesario para entender bases de código heredadas. Nuestra experiencia con múltiples clientes muestra que la comprensión asistida por GenAI de sistemas heredados es ahora una práctica estándar, más que un experimento. El esfuerzo de configuración varía, especialmente en enfoques avanzados como GraphRAG, y tiende a escalar con el tamaño y la complejidad de la base de código analizada. Aun así, el impacto en la productividad es constante y considerable. GenAI se ha convertido en una parte esencial de cómo exploramos y comprendemos sistemas heredados.

5. AGENTS.md

Probar

AGENTS.md es un formato común para proporcionar prompts a agentes de IA dedicados al desarrollo de software dentro de un proyecto. En esencia, funciona como un archivo README para agentes y no requiere campos ni formatos específicos más allá del uso de Markdown, confiando en la capacidad de los agentes de codificación basados en LLM para interpretar instrucciones escritas y legibles por humanos. Los usos típicos incluyen consejos sobre el uso de herramientas en el entorno de desarrollo, prompts de prueba y prácticas recomendadas para la gestión de commits. Aunque las herramientas de IA admiten varios métodos de ingeniería de contexto, the value of AGENTS.md lies in creating a simple convention for a file that acts as a starting point.

6. IA para migraciones de código

Probar

IA para migraciones de código abarca diversas formas de migración de código, desde reescrituras de lenguaje hasta actualizaciones de dependencias o frameworks— y rara vez son triviales, ya que suelen requerir meses de esfuerzo manual. Uno de nuestros equipos, al actualizar su versión de .NET framework, experimentó con el uso de IA para acortar el proceso. En el pasado, mencionamos OpenRewrite, una herramienta de refactorización determinista basada en reglas. El uso exclusivo de la IA para este tipo de actualizaciones ha demostrado ser, con frecuencia, costoso y propenso a generar conversaciones dispersas. En su lugar, el equipo combinó pipelines de actualización tradicionales con asistentes de codificación basados en agentes para gestionar transiciones complejas. En lugar de delegar toda la actualización, dividieron el proceso en pasos más pequeños y verificables: analizar errores de compilación, generar diferencias de migración y validar pruebas de forma iterativa. Este enfoque híbrido posiciona a los agentes de codificación con IA como colaboradores pragmáticos en el mantenimiento de software. Ejemplos en la industria, como la migración a gran escala de Google de int32 a int64, reflejan una tendencia similar. Aunque nuestros resultados son mixtos en cuanto al ahorro de tiempo medible, el potencial de IA para migraciones de código para reducir el trabajo repetitivo de las personas desarrolladoras es claro y merece seguir siendo explorado.

7. Delta Lake liquid clustering

Probar

Liquid clustering es una técnica para tablas de Delta Lake que actúa como una alternativa al particionamiento y al Z-ordering. Tradicionalmente, optimizar las tablas Delta para mejorar el rendimiento de lectura requería definir las claves de partición y Z-order al momento de crear la tabla, basándose en los patrones de consulta previstos. Modificar estas claves más adelante implicaba reescribir por completo los datos. En cambio, clustering utiliza un algoritmo basado en árboles para agrupar los datos según las claves designadas, las cuales pueden modificarse de forma incremental sin necesidad de reescribir toda la información. Esto ofrece mayor flexibilidad para admitir diversos patrones de consulta, reduciendo los costos de cómputo y mejorando el rendimiento de lectura.

Además, el Databricks Runtime para Delta Lake admite el liquid clustering automático que analiza los historiales de consultas, identifica las columnas óptimas y agrupa los datos en consecuencia. Tanto los usuarios de Delta Lake independiente como los del entorno Databricks Runtime pueden aprovechar esta técnica para optimizar el rendimiento de lectura.

8. Autoservicio de prototipado UI con GenAI

Probar

Usamos la expresión autoservicio de prototipado UI con GenAI para describir una técnica emergente en la que herramientas como Claude Code Figma Make, Miro AI y v0 permiten a las personas product managers generar prototipos interactivos y aptos para pruebas de usuario directamente a partir de indicaciones en texto. En lugar de crear manualmente los wireframes, los equipos pueden generar artefactos funcionales en HTML, CSS y JS en cuestión de minutos, ofreciendo la velocidad de un boceto pero con interactividad real y mayor fidelidad. Estos prototipos "desechables" sacrifican el detalle acabado por un aprendizaje rápido, lo que los hace ideales para la validación temprana durante los design sprints. Sin embargo, una mayor fidelidad puede generar un enfoque indebido en detalles menores o expectativas poco realistas sobre el esfuerzo de producción, por lo que es fundamental establecer un marco y expectativas claras.

Usada junto con la investigación de usuarios, esta técnica acelera la fase de descubrimiento al convertir ideas abstractas en experiencias tangibles que las personas usuarias pueden evaluar. No obstante, los equipos deben procurar que estas herramientas no sustituyan el proceso de investigación en sí. Bien aplicada, el autoservicio de prototipado acorta los ciclos de retroalimentación, reduce las barreras para quienes no son diseñadores y ayuda a los equipos a iterar con rapidez manteniendo un equilibrio saludable entre velocidad y calidad.

9. Salida estructurada de LLMs

Probar

La salida estructurada de LLMs es la práctica de restringir a un modelo de lenguaje grande para que genere respuestas en un formato predefinido, como JSON o una clase de programación específica. Esta técnica es esencial para crear aplicaciones confiables y listas para producción, transformando el texto típicamente impredecible del LLM en un contrato de datos determinista y legible por máquina. Basados en un uso exitoso en producción, movemos esta técnica de Assess a Trial.

Los enfoques van desde un simple formateo basado en prompts y model-native structured outputs hasta métodos de decodificación restringida más robustos mediante herramientas como Outlines e Instructor, que aplican máquinas de estados finitos para garantizar salidas válidas. Hemos utilizado con éxito esta técnica para extraer datos complejos y no estructurados de diversos tipos de documentos y convertirlos en JSON estructurado para su uso en la lógica de negocio posterior.

10. TCR (Test && Commit || Revert)

Probar

Test && commit || revert (TCR) es un flujo de trabajo de programación derivado del desarrollo guiado por pruebas (TDD) que promueve pasos muy pequeños y continuos mediante una regla simple: después de cada cambio, si las pruebas pasan, se hace commit de los cambios; si fallan, los cambios se revierten. Implementar TCR es sencillo: solo se necesita definir un script que automatice este ciclo dentro de la base de código. Originalmente introducido en un artículo canónico de Kent Beck, hemos comprobado que TCR refuerza buenas prácticas de codificación como YAGNI y KISS. Vale la pena evaluarlo mientras experimentamos con nuevos flujos de trabajo para construir software con GenAI.

11. Pruebas de IU impulsadas por IA

Evaluar

En el Radar anterior, las pruebas de IU impulsadas por IA (AI-powered UI testing) se centraban principalmente en las pruebas exploratorias donde señalamos que el carácter no determinista de los LLM podía introducir inestabilidad. Con el auge de MCP, ahora vemos que los principales frameworks para pruebas de interfaz de usuario como Playwright y Selenium han incorporado sus propios servidores MCP (playwright-mcp, mcp-selenium). Estos permiten una automatización de navegador más confiable mediante sus tecnologías nativas lo que facilita que los asistentes de codificación generen pruebas de interfaz de usuario estables en Playwright o Selenium. Aunque las pruebas de interfaz de usuario con IA siguen siendo un campo en rápida evolución (la versión más reciente de Playwright, por ejemplo, introdujo Playwright Agents nos entusiasman estos avances y esperamos ver más orientación práctica y experiencias de campo en el futuro cercano.

12. Anclar los agentes de codificación a una aplicación de referencia

Evaluar

En el pasado, mencionamos el patrón de plantillas de servicio a la medida que ayudó a las organizaciones que adoptan microservicios al proporcionar configuraciones predeterminadas razonables para crear nuevos servicios e integrarlos sin problemas con la infraestructura existente. Con el tiempo, sin embargo, la divergencia de código entre estas plantillas y los servicios existentes tiende a aumentar a medida que surgen nuevas dependencias, frameworks y patrones arquitectónicos. Para mantener buenas prácticas y coherencia arquitectónica, especialmente en la era de los agentes de codificación hemos estado experimentando con anclar los agentes de codificación a una aplicación de referencia. Este patrón guía a los agentes de código generativo al proporcionar una aplicación de referencia viva y compilable en lugar de ejemplos estáticos en los prompts. Un servidor Model Context Protocol (MCP) expone tanto el código de la plantilla de referencia como los diffs de los commits, lo que permite a los agentes detectar divergencias y proponer correcciones. Este enfoque transforma las plantillas estáticas en planos vivos y adaptables que la IA puede consultar de forma inteligente, manteniendo la coherencia, reduciendo la divergencia y mejorando el control sobre el scaffolding impulsado por IA a medida que los sistemas evolucionan.

13. Ingeniería de contexto

Evaluar

Ingeniería de contexto es el diseño y la optimización sistemática de la información proporcionada a un modelo de lenguaje grande (LLM, por sus siglas en inglés) durante la inferencia, con el fin de producir de manera confiable el resultado deseado. Implica estructurar, seleccionar y secuenciar los elementos contextuales como prompts, datos recuperados, memoria, prompts y señales del entorno para que las capas internas del modelo operen en un estado óptimo. A diferencia de la ingeniería de prompts, que se enfoca únicamente en la redacción de los mismos, la ingeniería de contexto considera toda la configuración del contexto: cómo se organiza y entrega el conocimiento relevante, los prompts y el contexto previo para lograr los resultados más efectivos. Actualmente, se utilizan una variedad de técnicas que pueden agruparse en tres áreas: configuración del contexto, que abarca tácticas de curación como el uso de prompts del sistema mínimos, ejemplos few-shot canónicos y herramientas eficientes en tokens para acciones decisivas; gestión del contexto para tareas de largo horizonte con ventanas de contexto finitas mediante resumen de contexto, toma de notas estructurada para mantener memorias externas y arquitecturas de subagentes que aíslan y resumen subtareas

complejas; y recuperación dinámica de información que depende de la recuperación de contexto justo a tiempo (JIT), donde los agentes cargan datos externos de forma autónoma sólo cuando son inmediatamente relevantes, maximizando la eficiencia y la precisión.

14. GenAI para ingeniería progresiva

Evaluar

GenAI para ingeniería progresiva es una técnica emergente para modernizar sistemas heredados mediante descripciones generadas por inteligencia artificial sobre bases de código existentes. Introduce una etapa explícita centrada en qué hace el código heredado (su especificación), ocultando deliberadamente cómo está implementado actualmente. Está relacionada con el spec-driven development, aunque se aplica específicamente a la modernización de sistemas heredados. Al generar e iterar sobre descripciones funcionales antes de reescribir el código, los equipos pueden usar GenAI para revelar lógica oculta, dependencias y casos límite que de otro modo podrían pasarse por alto.

Enfocarse en el espacio del problema en lugar del sistema existente también permite que los modelos de GenAI exploren soluciones más creativas y orientadas al futuro. El flujo de trabajo sigue un ciclo de ingeniería inversa → diseño/solución → ingeniería progresiva, lo que permite que tanto las personas como los agentes de IA razonen a un nivel más alto antes de comprometerse con una implementación.

En Thoughtworks, estamos viendo a varios equipos aplicar con éxito este enfoque para acelerar la reescritura de sistemas heredados. El objetivo no es ocultar por completo los detalles de implementación, sino introducir una abstracción temporal que ayude a los equipos y agentes a explorar alternativas sin estar limitados por la estructura actual. Esta técnica está mostrando resultados prometedores al generar código más limpio, mantenible y preparado para el futuro, al tiempo que reduce el esfuerzo necesario para comprender las implementaciones existentes.

15. GraphQL como patrón de acceso a datos para LLMs

Evaluar

GraphQL como patrón de acceso a datos para LLMs es un enfoque emergente para crear una capa de acceso a datos uniforme y compatible con modelos, que mejora la ingeniería de contexto. Permite a los equipos exponer datos estructurados y consultables sin otorgar a los modelos acceso directo a las bases de datos. A diferencia de las API REST, que tienden a recuperar datos en exceso o requieren nuevos endpoints o filtros para cada caso de uso, GraphQL permite que el modelo obtenga solo los datos que necesita, reduciendo el ruido, mejorando la relevancia del contexto y disminuyendo el uso de tokens.

Un esquema GraphQL bien definido también proporciona metadatos que los LLM pueden usar para razonar sobre las entidades y relaciones disponibles, lo que permite consultas dinámicas con reconocimiento de esquema para casos de uso con agentes. Este patrón ofrece un punto intermedio seguro entre REST y SQL, equilibrando los controles de gobernanza con un acceso flexible.

Sin embargo, el enfoque depende de esquemas bien estructurados y nombres de campo significativos. Interpretar la semántica de los esquemas y navegar por estructuras complejas sigue siendo un desafío, y lo que resulta difícil de razonar para las personas suele ser igual de difícil para los LLM. También es importante tener en cuenta los vectores adicionales para ataques DoS, así como los desafíos habituales de GraphQL, como el almacenamiento en caché y el versionado.

16. Flujos de conocimiento sobre reservas de conocimiento

Evaluar

Una pregunta que recibimos con frecuencia es ¿Cómo podemos mejorar la forma en que compartimos información entre nuestros equipos?. Las técnicas para gestionar el conocimiento organizacional siguen evolucionando y una perspectiva que hemos encontrado valiosa proviene del pensamiento sistémico: los conceptos de flujos de conocimiento y reservas de conocimiento. Originalmente provenientes de la economía, este enfoque invita a los equipos a ver su conocimiento organizacional como un sistema, donde las reservas representan el conocimiento acumulado y los flujos representan cómo ese conocimiento se mueve y evoluciona dentro de la organización. Aumentar el flujo de conocimiento externo hacia una organización tiende a impulsar la innovación. Una forma comprobada de mejorar este flujo es establecer comunidades de práctica, que de manera consistente muestran beneficios medibles. Otra es buscar deliberadamente fuentes de conocimiento diversas y externas. A medida que las herramientas de GenAI hacen que las reservas de conocimiento existentes sean más accesibles, vale la pena recordar que fomentar ideas frescas y perspectivas externas es tan importante como adoptar nuevas tecnologías.

17. LLM como juez

Evaluar

Usar un LLM como juez para evaluar la salida de otro sistema, generalmente un productor basado en LLM, ha ganado atención por su potencial de ofrecer una evaluación automatizada y escalable en inteligencia artificial generativa. Sin embargo, movemos este tema de Trial a Assess para reflejar las nuevas complejidades y riesgos identificados. Si bien esta técnica ofrece velocidad y escala, a menudo falla como sustituto confiable del juicio humano. Las evaluaciones son propensas a sesgos de posición, sesgos de verbosidad y baja robustez. Un problema más grave es la contaminación por escala: Cuando se utiliza LLM como juez en pipelines de entrenamiento para el modelado de recompensas, puede introducir sesgos de autoafirmación, donde una familia de modelos favorece sus propias salidas y filtraciones de preferencias, difuminando la frontera entre entrenamiento y prueba. Estos defectos han generado resultados que inflan las métricas de rendimiento sin validez en el mundo real. Algunos estudios de investigación han realizado análisis más rigurosos sobre este patrón. Para contrarrestar estos problemas, estamos explorando técnicas mejoradas, como el uso de LLMs como jurado ((empleando varios modelos para llegar a un consenso) o el razonamiento en cadena (chain-of-thought o CoT) durante la evaluación. Si bien estos métodos buscan aumentar la fiabilidad, también incrementan el costo y la complejidad. Recomendamos a los equipos tratar esta técnica con precaución, asegurando verificación humana, transparencia y supervisión ética antes de incorporar LLMs como jueces en flujos de trabajo críticos. El enfoque sigue siendo potente, pero menos maduro de lo que se creía.

18. La recuperación de información en el dispositivo

Evaluar

La recuperación de información en el dispositivo es una técnica que permite ejecutar búsquedas, reconocimiento de contexto y generación aumentada por recuperación (RAG) directamente en los dispositivos del usuario —ya sean móviles, de escritorio o de borde (edge devices)— priorizando la privacidad y la eficiencia computacional. Esta técnica combina una base de datos local ligera con un modelo optimizado para inferencia en el propio dispositivo. Una implementación prometedora combina sqlite-vec, una extensión de SQLite que habilita búsquedas vectoriales dentro de la base de datos embebida, con EmbeddingGemma, un modelo de embeddings de 300 millones de parámetros construido sobre la arquitectura Gemma 3. Optimizada para la eficiencia y entornos con

recursos limitados, esta combinación mantiene los datos cerca del borde –es decir, en los propios dispositivos– reduciendo la dependencia de las API en la nube y mejorando la latencia y la privacidad. Recomendamos que los equipos evalúen esta técnica para aplicaciones local-first y otros casos donde la soberanía de los datos, la baja latencia y la privacidad sean factores críticos.

19. SAIF

Evaluar

SAIF (Secure AI Framework) es un framework desarrollado por Google que ofrece una guía práctica para gestionar los riesgos de seguridad en la inteligencia artificial. Aborda de forma sistemática amenazas comunes como la manipulación de datos (data poisoning) y la inyección de prompts, mediante un mapa de riesgos claro y análisis de componentes. SAIF también proporciona estrategias prácticas de mitigación para cada una de estas amenazas. Consideramos que su enfoque en los riesgos emergentes relacionados con la construcción de sistemas con agentes resulta especialmente oportuno y valioso. SAIF ofrece un manual conciso y accionable que los equipos pueden utilizar para fortalecer sus prácticas de seguridad en el uso de LLMs y aplicaciones impulsadas por IA.

20. Service mesh sin sidecar

Evaluar

Ya que aún persisten los costos y la complejidad operativa de las mallas de servicios (service meshes) basados en sidecars, nos entusiasma ver otra opción surgir para malla de servicios sin sidecar: Istio ambient mode. El ambient mode introduce una arquitectura en capas que separa las responsabilidades entre dos componentes clave: el proxy L4 por nodo (ztunnel) y el proxy L7 por espacio de nombres (Waypoint proxy). ztunnel garantiza que el tráfico L3 y L4 se transporte de manera eficiente y segura. Este hace funcionar el plano de datos ambient, obteniendo certificados para todas las identidades de nodo y, a su vez, gestiona la redirección del tráfico hacia y desde las cargas de trabajo habilitadas para ambient. El proxy Waypoint, un componente opcional del modo ambient, habilita características más avanzadas de Istio como gestión de tráfico, seguridad y observabilidad. Hemos tenido buenas experiencias con este modo en clústeres de pequeña escala y esperamos obtener más información y mejores prácticas a medida que crece su adopción.

21. Small language models

Evaluar

Hemos observado un progreso constante en el desarrollo de small language models (SLMs) a lo largo de varios volúmenes del Radar Tecnológico. Con el creciente interés en la creación de soluciones agénticas, estamos viendo cada vez más evidencia de que los SLMs pueden impulsar la IA agéntica de manera eficiente. La mayoría de los flujos de trabajo basados en agentes actuales se centran en tareas específicas y repetitivas que no requieren razonamiento avanzado, lo que los convierte en una buena opción para los SLMs. Los avances continuos en modelos como Phi-3, SmolLM2 y DeepSeek sugieren que los SLMs ofrecen suficiente capacidad para este tipo de tareas, con beneficios adicionales de menor costo, menor latencia y menor consumo de recursos en comparación con los LLMs. Vale la pena considerar los SLMs como la opción predeterminada para los flujos de trabajo basados en agentes, reservando a los LLMs más grandes y con mayor consumo de recursos sólo para cuando sea necesario.

22. Spec-driven development

Evaluar

Spec-driven development es un enfoque emergente para los flujos de trabajo de codificación asistida por IA. Aunque la definición del término aún está evolucionando, generalmente se refiere a flujos de trabajo que comienzan con una especificación funcional estructurada y luego avanzan a través de varios pasos para descomponerla en partes más pequeñas, soluciones y tareas. La especificación puede adoptar muchas formas: un único documento, un conjunto de documentos o artefactos estructurados que capturan diferentes aspectos funcionales.

Hemos visto a muchas personas desarrolladoras adoptar este estilo (e incluso tenemos una versión propia que compartimos internamente en Thoughtworks). Tres herramientas en particular han explorado recientemente interpretaciones distintas de spec-driven development. Kiro de Amazon guía a las personas usuarias a través de tres etapas de flujo de trabajo: requisitos, diseño y creación de tareas. GitHub's spec-kit de GitHub sigue un proceso similar de tres pasos, pero añade una orquestación más rica, prompts configurables y una "constitución" que define principios inmutables que siempre deben cumplirse. Tessl Framework (aún en beta privada en septiembre de 2025) adopta un enfoque más radical, en el que la propia especificación se convierte en el artefacto mantenido, en lugar del código.

Encontramos este espacio fascinante, aunque los flujos de trabajo siguen siendo elaborados y prescriptivos. Estas herramientas se comportan de manera muy diferente según el tamaño y tipo de tarea; algunas generan archivos de especificación extensos y difíciles de revisar, y cuando producen PRDs o historias de usuario, a veces no queda claro para quién están dirigidos. Puede que estemos reaprendiendo una bitter lesson elaborar reglas detalladas a mano para la IA, en última instancia, no es escalable.

23. Equipo de agentes de codificación

Evaluar

El equipo de agentes de codificación se refiere a una técnica en la que una persona desarrolladora orquesta varios agentes de codificación con IA, cada uno con un rol distinto (como arquitecto, especialista en back-end, tester) para colaborar en una tarea de desarrollo. Esta práctica cuenta con el respaldo de herramientas como Claude Code, Roo Code y Kilo Code que permiten el uso de subagentes y múltiples modos de operación. Basándose en el principio comprobado de que asignar roles y perfiles específicos a los LLM mejora la calidad de los resultados, la idea es obtener mejores resultados al coordinar varios agentes con roles definidos, en lugar de depender de uno solo de propósito general. El nivel óptimo de granularidad de los agentes aún se está explorando, pero puede ir más allá de una simple correspondencia uno a uno con los roles tradicionales de un equipo. Este enfoque marca una transición hacia pipelines de desarrollo asistidos por IA, orquestados y de múltiples etapas.

24. Programación consciente de la topología

Evaluar

Las GPU y LPU han dejado de ser dispositivos independientes para convertirse en redes estrechamente acopladas de aceleradores cuyo rendimiento depende de la ubicación y la topología. En sistemas a escala de rack como el NVL72 de NVIDIA, 72 GPUs comparten más de 13 TB de VRAM y funcionan como un único acelerador, hasta que las cargas de trabajo cruzan las islas de conmutadores, convirtiendo las operaciones colectivas en cuellos de botella. De manera similar, la arquitectura de Groq (planificada por software en tiempo de compilación) asume un movimiento de datos determinista; una planificación aleatoria rompe esas suposiciones y su previsibilidad. Incluso

dentro de un mismo centro de datos, el rendimiento de las GPU puede variar significativamente, generando la necesidad de una programación consciente de la topología que considere tanto el diseño del hardware como su variabilidad al asignar trabajos.

Los planificadores ingenuos que ignoran la topología de NVLink, PCIe o NIC suelen distribuir cargas de trabajo multi-GPU de forma arbitraria, lo que provoca una degradación en los tiempos de ejecución y la eficiencia. Las cargas de entrenamiento, que son síncronas y dependientes del ancho de banda, se benefician de islas NVLink contiguas con rutas uniformes y de alta velocidad para las etapas de all-reduce y pipeline. Estos trabajos deben programarse en función del ancho de banda del tejido, evitando saltos entre conmutadores y tratando los límites de enlace, conmutador y nodo como dominios de falla. Las cargas de inferencia, por el contrario, están limitadas por la latencia y los SLO, y suelen equilibrar la replicación para alta disponibilidad entre dominios, combinándola con particionamiento (sharding) para mantener la localidad de los mixture of experts (MoE) y las cachés KV en las rutas más cortas. Optimizar la ubicación para las fases de prefill y decode, el microprocesamiento por lotes (micro-batching) y el aislamiento de inquilinos mejora aún más la eficiencia. Creemos que la programación consciente de la topología se volverá esencial a medida que el rendimiento de los aceleradores dependa cada vez más de la red y la topología del centro de datos. Nuestros equipos ya están evaluando Kueue y proyectos relacionados para mejorar la precisión en la asignación, incrementar el rendimiento y garantizar una escalabilidad confiable para nuestros clientes.

25. Análisis de flujo tóxico para IA

Evaluar

El ya conocido chiste de que la S en MCP significa “seguridad” oculta un problema muy real. Cuando los agentes se comunican entre sí, ya sea mediante la invocación de herramientas o llamadas a API, pueden encontrarse rápidamente con lo que se ha denominado la tríada letal: acceso a datos privados, exposición a contenido no confiable y capacidad de comunicación externa. Los agentes que combinan estos tres elementos son altamente vulnerables. Dado que los LLMs tienden a seguir las instrucciones incluidas en su entrada, cualquier contenido que contenga una orden para exfiltrar datos a una fuente no confiable puede generar fácilmente filtraciones de datos. Una técnica emergente para mitigar este riesgo es el análisis de flujo tóxico, (toxic flow analysis), que examina el grafo de flujo de un sistema de agentes para identificar posibles rutas de datos inseguras que requieren una investigación más profunda. Aunque todavía se encuentra en una etapa temprana, el análisis de flujo tóxico representa una de las aproximaciones más prometedoras para detectar los nuevos vectores de ataque a los que los sistemas de agentes y los servidores MCP están cada vez más expuestos.

26. TI en la sombra acelerado por IA

Resistir

La IA está reduciendo las barreras para que quienes no codifican puedan crear e integrar software por sí mismos, en lugar de esperar a que el departamento de TI se encargue de sus necesidades o requisitos. Aunque nos entusiasma el potencial que esto desbloquea, también nos preocupan los primeros indicios de TI en la sombra acelerado por IA. Las plataformas de automatización de flujos de trabajo sin código ahora admiten la integración de APIs de IA (por ejemplo, de OpenAI o Anthropic), lo que hace tentador usar la IA como cinta adhesiva — haciendo integraciones que antes no eran posibles mediante IA, como convertir mensajes de chat de un sistema en llamadas a la API de un ERP. Al mismo tiempo, los asistentes de codificación impulsados por IA se están volviendo más autónomos,

lo que permite a quienes no codifican, pero que tienen una formación básica, la posibilidad de crear aplicaciones de utilidad internas.

Esto tiene todas las características de la próxima evolución de las hojas de cálculo que aún impulsan procesos críticos en algunas empresas — pero con un alcance mucho mayor. Si no se controla, esta nueva TI en la sombra podría provocar la proliferación de aplicaciones no gobernadas y potencialmente inseguras, dispersando los datos en cada vez más sistemas. Las organizaciones deben ser conscientes de estos riesgos y evaluar cuidadosamente las ventajas y desventajas entre la rápida resolución de problemas y la estabilidad a largo plazo.

27. Capacity-driven development

Resistir

La clave del éxito de las prácticas modernas de desarrollo de software es mantener el foco en el flujo de trabajo. Los equipos alineados al flujo se concentran en un único flujo de valor, como un recorrido de usuario o un producto, lo que les permite entregar valor de extremo a extremo de manera eficiente. Sin embargo, estamos observando una tendencia preocupante hacia el capacity-driven development, dónde equipos alineados de esta forma asumen características o funcionalidades de otros productos o flujos cuando tienen capacidad disponible. Aunque esto puede parecer eficiente a corto plazo, representa una optimización local más adecuada para manejar picos repentinos de demanda. Cuando se normaliza, aumenta la carga cognitiva y la deuda técnica, y en el peor de los casos puede provocar un colapso por congestión, ya que el costo del cambio de contexto entre productos se acumula. Los equipos con capacidad disponible obtienen mejores resultados al centrarse en mejorar la salud del sistema. Para gestionar la capacidad de manera efectiva, utiliza límites al trabajo en progreso (WIP) para controlar el trabajo entre flujos adyacentes, considera la capacitación cruzada para equilibrar los períodos de alta demanda y aplica técnicas de dynamic reteaming cuando sea necesario.

28. Complacencia con el código generado por IA

Resistir

A medida que los asistentes y agentes de codificación con IA ganan terreno, también aumenta la cantidad de datos e investigaciones que señalan preocupaciones sobre la complacencia con el código generado por IA. Si bien existe amplia evidencia de que estas herramientas pueden acelerar el desarrollo (especialmente en proyectos de prototipado y greenfield), los estudios demuestran que la calidad del código puede deteriorarse con el tiempo. La investigación de GitClear (2024) reveló que el código duplicado y el code churn han aumentado más de lo esperado, mientras que la actividad de refactorización en los historiales de commits ha disminuido. En una tendencia similar, una investigación Microsoft sobre trabajadores del conocimiento muestra que la confianza impulsada por la IA suele darse a costa del pensamiento crítico, un patrón que también hemos observado cuando se afianza la complacencia tras el uso prolongado de asistentes de codificación.

El auge de estos agentes amplifica aún más estos riesgos, ya que la IA ahora genera conjuntos de cambios más grandes y difíciles de revisar. Como en cualquier sistema, acelerar una parte del flujo de trabajo aumenta la presión sobre las demás. Nuestros equipos han comprobado que usar la IA de forma efectiva en entornos de producción requiere un enfoque renovado en la calidad del código. Recomendamos reforzar prácticas consolidadas como TDD y el análisis estático, e integrarlas directamente en los flujos de trabajo de codificación, por ejemplo, mediante prompts compartidos seleccionados para equipos de software.

29. Conversión ingenua de API a MCP

Resistir

Las organizaciones están cada vez más interesadas en permitir que los agentes de IA interactúen con sus sistemas existentes, a menudo intentando una conversión directa y sin fricciones de las API internas al Model Context Protocol (MCP). Un número creciente de herramientas, como MCP link y FastAPI-MCP, buscan facilitar este proceso.

Aconsejamos evitar esta conversión ingenua de API a MCP. Las API suelen estar diseñadas para desarrolladores humanos y a menudo consisten en acciones granulares y atómicas que, al ser encadenadas por una IA, pueden generar un uso excesivo de tokens, contaminación del contexto y bajo rendimiento de los agentes. Además, estas API (especialmente las internas) suelen exponen datos sensibles o permiten operaciones destructivas. Para los desarrolladores humanos, esos riesgos se mitigan mediante patrones de arquitectura y revisiones de código; sin embargo, cuando las API se exponen de forma ingenua a los agentes a través de MCP, no existe una forma confiable ni determinista de evitar que un agente de IA autónomo haga un uso indebido de estos endpoints. Recomendamos diseñar un servidor MCP dedicado y seguro, específicamente adaptado a los flujos de trabajo agéntico, construido sobre las API existentes.

30. Equipos de ingeniería de datos separados

Resistir

Organizar equipos de ingeniería de datos separados para desarrollar y poseer pipelines y productos de datos, independientes de los dominios de negocio alineados al flujo que atienden, es un antipatrón que genera ineficiencias y resultados de negocio débiles. Esta estructura repite errores del pasado, como aislar las funciones de DevOps, testing o despliegue creando silos de conocimiento, cuellos de botella y esfuerzos desperdiciados. Sin una colaboración estrecha, las personas ingenieras de datos suelen carecer del contexto de negocio y del dominio necesarios para diseñar productos de datos significativos, lo que limita tanto su adopción como su valor. En contraste, los equipos de plataformas de datos deberían centrarse en mantener la infraestructura compartida, mientras que los equipos de negocio multifuncionales construyen y poseen sus propios productos de datos, siguiendo los principios de data mesh Colocamos esta práctica en Hold para desincentivar con firmeza los patrones organizativos en silos, especialmente ahora que la necesidad de datos ricos en contexto y listos para IA sigue creciendo.

31. Text to SQL

Resistir

Text to SQL utiliza LLMs para traducir lenguaje natural en SQL ejecutable, pero su confiabilidad suele estar por debajo de lo esperado. Hemos movido este blip a Hold para desalentar su uso en flujos de trabajo no supervisados, por ejemplo, al convertir dinámicamente consultas generadas por usuarios cuando el resultado se oculta o se ejecuta de forma automática. En estos casos, los LLMs suelen alucinar debido a una comprensión limitada del esquema o del dominio, lo que puede provocar recuperaciones de datos incorrectas o modificaciones no intencionadas. La naturaleza no determinista de las salidas de los LLMs también dificulta depurar y auditar errores.

Aconsejamos usar Text to SQL con precaución, Se debe exigir revisión humana de las consultas generadas. Para inteligencia de negocio agéntica, evita el acceso directo a bases de datos y utiliza en su lugar una capa semántica de abstracción de datos gobernada, como Cube o la capa semántica de dbt's o bien una capa de acceso semánticamente rica como GraphQL o MCP.

Plataformas

Adoptar

32. Arm en la nube

Probar

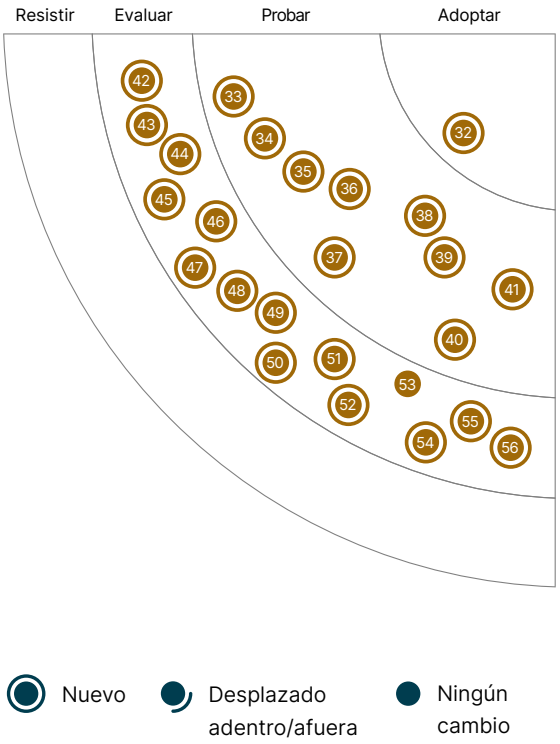
33. Apache Paimon
34. DataDog LLM Observability
35. Delta Sharing
36. Dovetail
37. Langdock
38. LangSmith
39. Model Context Protocol (MCP)
40. n8n
41. OpenThread

Evaluar

42. AG-UI Protocol
43. Agent-to-Agent (A2A) Protocol
44. Amazon S3 Vectors
45. Ardoq
46. CloudNativePG
47. Coder
48. Graft
49. groundcover
50. Karmada
51. OpenFeature
52. Oxide
53. Restate
54. SkyPilot
55. StarRocks
56. Uncloud

Resistir

—



Nuevo

Desplazado adentro/afuera

Ningún cambio

32. Arm en la nube

Adoptar

Las instancias de computación Arm en la nube se han vuelto cada vez más populares en los últimos años por su eficiencia en costos y energía en comparación con las instancias tradicionales basadas en x86. Los principales proveedores de nube, entre ellos AWS, Azure y GCP ahora ofrecen opciones robustas basadas en Arm. Estas instancias resultan especialmente atractivas para cargas de trabajo a gran escala o sensibles a costos. Muchos de nuestros equipos han migrado con éxito cargas de trabajo como microservicios, bases de datos de código abierto e incluso computación de alto rendimiento hacia Arm con cambios mínimos en el código y sólo pequeños ajustes en los scripts de compilación. Las nuevas aplicaciones y sistemas basados en la nube adoptan cada vez más Arm en la nube como configuración predeterminada. Según nuestra experiencia, recomendamos las instancias de cómputo Arm para la mayoría de las cargas de trabajo, a menos que existan dependencias específicas de arquitectura. Las herramientas modernas, como las multi-arch Docker images, simplifican aún más la creación y el despliegue en entornos tanto Arm como x86.

33. Apache Paimon

Probar

Apache Paimon es un formato de data lake de código abierto diseñado para habilitar la arquitectura lakehouse. Se integra de forma fluida con motores de procesamiento como arquitectura lakehouse. ofreciendo soporte tanto para operaciones en streaming como por lotes (batch). Una de las principales ventajas de la arquitectura de Paimon radica en su combinación de un formato estándar de data lake con una Flink y Spark, supporting both streaming and batch operations. A key advantage of Paimon's architecture lies in its fusion of a standard data lake format with an estructura LSM (log-structured merge-tree). Esta combinación resuelve los desafíos tradicionales de las actualizaciones de alto rendimiento y las lecturas de baja latencia en los data lakes.

Paimon admite tablas con clave primaria para actualizaciones en tiempo real y de alto rendimiento, e incluye un motor de fusión personalizable para deduplicación, actualizaciones parciales y agregaciones. Este diseño hace posible una ingesta eficiente de datos en streaming y la gestión de estado mutable directamente dentro del lake. Además, Paimon también ofrece capacidades maduras de data lake, como metadatos escalables, transacciones ACID, time travel, evolución de esquemas y diseños de datos optimizados mediante compresión y Z-ordering. Recomendamos evaluar Paimon para proyectos que necesiten una capa de almacenamiento unificada capaz de manejar eficientemente datos masivos de solo anexado(append-only) a gran escala y actualizaciones complejas en streaming en tiempo real.

34. DataDog LLM Observability

Probar

Datadog LLM Observability proporciona trazabilidad, monitoreo y diagnóstico de extremo a extremo para modelos grandes de lenguaje y flujos de trabajo de aplicaciones agénticas. Mapea cada prompt, llamada de herramienta y paso intermedio en spans y traces; supervisa la latencia, el uso de tokens, los errores y las métricas de calidad; y se integra con la suite más amplia de APM y observabilidad de Datadog.

Las organizaciones que ya utilizan Datadog y están familiarizadas con su estructura de costos pueden encontrar que la función de observabilidad para LLM es una forma directa de obtener visibilidad sobre las cargas de trabajo de IA, siempre que estas puedan instrumentarse. Sin embargo, la configuración y el uso de la instrumentación para LLMs requiere cuidado y una comprensión sólida tanto de las cargas de trabajo como de su implementación. Recomendamos que los equipos de ingeniería de

datos y operaciones trabajen en estrecha colaboración al implementarla. Consulta también nuestra recomendación sobre evitar equipos de ingeniería de datos separados.

35. Delta Sharing

Probar

Delta Sharing es un estándar y protocolo abierto para el intercambio seguro de datos entre plataformas, desarrollado por Databricks y la Linux Foundation. Es independiente de la nube, lo que permite a las organizaciones compartir datos en tiempo real entre distintos proveedores de nube y entornos locales sin necesidad de copiarlos ni replicarlos, preservando su frescura y eliminando costos de duplicación. Hemos visto a una empresa de comercio electrónico utilizar Delta Sharing con éxito para reemplazar un sistema fragmentado de intercambio de datos con socios por una plataforma centralizada, segura y en tiempo real, mejorando significativamente la colaboración. El protocolo usa una sencilla API REST para emitir URL prefirmadas de corta duración, lo que permite a los destinatarios recuperar grandes conjuntos de datos mediante herramientas como pandas, Spark o Power BI. Admite compartir tablas de datos, vistas, modelos de IA y notebooks. Aunque ofrece una sólida gobernanza centralizada y capacidades de auditoría, los usuarios deben tener en cuenta los costos de salida de datos en la nube, que pueden convertirse en un riesgo operativo importante si no se gestionan adecuadamente.

36. Dovetail

Probar

Dovetail platform aborda el desafío persistente de gestionar datos de investigación cualitativa fragmentados. Proporciona un repositorio centralizado para entrevistas con usuarios, transcripciones e insights, convirtiendo los datos en bruto en un activo estructurado y analizable. Lo hemos encontrado muy valioso en los flujos de trabajo de descubrimiento de producto, especialmente para crear un rastro de evidencias que vinculan citas de clientes y temas sintetizados directamente con hipótesis de producto y ROI estimado. Al hacerlo, Dovetail refuerza el papel de los datos cualitativos en la toma de decisiones de producto.

37. Langdock

Probar

Langdock es una plataforma para que las organizaciones desarrollen y ejecuten agentes y flujos de trabajo de inteligencia artificial generativa para operaciones internas. Proporciona un entorno unificado con asistentes de chat internos, una capa de API para conectarse a múltiples LLMs y herramientas para crear flujos de trabajo basados en agentes que se integran con sistemas como Slack, Confluence y Google Drive. La plataforma pone énfasis en la soberanía de los datos, ofreciendo opciones on-premise y alojadas en la Unión Europea, con estándares de cumplimiento empresarial.

Las organizaciones que implementen Langdock deben seguir prestando especial atención a la gobernanza de datos y aplicar técnicas como el análisis de flujo tóxico para evitar la tríada letal. Quienes adopten la plataforma también deben considerar su nivel de madurez, evaluar las integraciones específicas que necesitan y planificar cualquier desarrollo personalizado que pueda ser necesario.

38. LangSmith

Probar

LangSmith es una plataforma alojada desarrollada por el equipo de LangChain que ofrece observabilidad, trazabilidad y evaluación para aplicaciones basadas en LLMs. Permite capturar trazas detalladas de cadenas, herramientas y prompts, lo que facilita depurar y analizar el comportamiento de los modelos, identificar regresiones de rendimiento y gestionar conjuntos de datos de evaluación. LangSmith es un SaaS propietario con soporte limitado para flujos de trabajo fuera del ecosistema de LangChain, por lo que resulta más atractivo para los equipos que ya utilizan esta plataforma. Su soporte integrado para la evaluación y experimentación de prompts es considerablemente más completo que el de alternativas de código abierto como Langfuse.

39. Model Context Protocol (MCP)

Probar

El Model Context Protocol (MCP) es un estándar abierto que define cómo las aplicaciones y agentes basados en LLMs se integran con fuentes de datos y herramientas externas, mejorando significativamente la calidad de las salidas generadas por IA. MCP se centra en el acceso al contexto y a las herramientas, diferenciándose del Agent2Agent (A2A) que regula la comunicación entre agentes. Específica servidores (para datos y herramientas como bases de datos, wikis y servicios) y clientes (agentes, aplicaciones y asistentes de código). Desde su último blip, la adopción de MCP ha crecido rápidamente, con grandes empresas como JetBrains (IntelliJ) y Apple uniéndose al ecosistema, junto con frameworks emergentes como FastMCP. Un estándar preliminar del registro MCP ahora permite la detección de herramientas públicas y propietarias. Sin embargo, la rápida evolución de MCP también ha introducido brechas arquitectónicas, lo que ha generado críticas por pasar por alto buenas prácticas establecidas de RPC. Para aplicaciones en producción, los equipos deben mirar más allá del revuelo y aplicar un análisis adicional, mitigando flujos tóxicos con herramientas como MCP-Scan y monitoreando de cerca el módulo de autorización en borrador para garantizar la seguridad.

40. n8n

Probar

n8n es una plataforma de automatización de flujos de trabajo con licencia fair-code, similar a Zapier o Make (anteriormente Integromat), pero creada para desarrolladores que buscan una opción autoalojada, extensible y controlable mediante código. Ofrece un enfoque visual y de bajo código para la creación de flujos de trabajo, más simple que Apache Airflow, pero manteniendo compatibilidad con código personalizado en JavaScript o Python.

Su principal caso de uso es integrar múltiples servicios dentro de flujos automatizados, aunque también puede conectar LLMs con fuentes de datos, memoria y herramientas configurables. Muchos de nuestros equipos utilizan n8n para crear rápidamente prototipos de flujos de trabajo basados en agentes que se activan desde aplicaciones de chat o webhooks, aprovechando con frecuencia sus funciones de importación y exportación para generar flujos con asistencia de IA. Como siempre, recomendamos precaución al utilizar plataformas de bajo código en entornos de producción. Sin embargo, el hecho de que n8n sea autoalojable y que los flujos pueden definirse mediante código ayuda a mitigar algunos de esos riesgos.

41. OpenThread

Probar

OpenThread es una implementación de código abierto del protocolo de red Thread desarrollada por Google. Admite todas las características clave de la especificación Thread, incluyendo capas de red como IPv6, 6LoWPAN y LR-WPAN, así como capacidades de red en malla que permiten que un dispositivo funcione tanto como nodo como enrutador de frontera. OpenThread se ejecuta en una amplia variedad de plataformas de hardware, aprovechando una capa de abstracción flexible y de puntos de integración que permiten a los fabricantes incorporar sus propias capacidades de radio y cifrado. Este protocolo maduro se utiliza ampliamente en productos comerciales y, en nuestra experiencia, ha demostrado ser confiable para crear soluciones de IoT diversas, desde dispositivos de bajo consumo alimentados por batería hasta grandes redes malladas de sensores.

42. AG-UI Protocol

Evaluar

AG-UI es un protocolo abierto y una biblioteca diseñados para estandarizar la comunicación entre interfaces de usuario avanzados y agentes. Enfocado a agentes que interactúan directamente con usuarios finales, utiliza un middleware e integraciones de cliente para generalizar en cualquier frontend y backend. El protocolo define una forma coherente para que los agentes del backend se comuniquen con las aplicaciones del frontend, permitiendo una colaboración en tiempo real y con estado entre la IA y los usuarios. Es compatible con múltiples protocolos de transporte, incluidos SSE y WebSockets, y proporciona tipos de eventos estandarizados para representar diferentes estados de ejecución del agente. Ofrece soporte integrado para frameworks de agentes populares como LangGraph y Pydantic AI, con integraciones de la comunidad para otros.

43. Agent-to-Agent (A2A) Protocol

Evaluar

Agent2Agent (A2A) es un protocolo que define un estándar para la comunicación e interacción entre agentes en flujos de trabajo complejos y multiagente. Utiliza Agent Cards para describir los elementos clave de la comunicación entre agentes, incluyendo el descubrimiento de habilidades y la especificación de esquemas de transporte y seguridad. A2A complementa el Model Context Protocol (MCP) (MCP) al centrarse en la comunicación entre agentes sin exponer detalles internos como el estado, la memoria o los procesos internos de un agente. El protocolo promueve buenas prácticas como un enfoque asíncrono por defecto para tareas de larga duración, respuestas en streaming para actualizaciones incrementales y transporte seguro con HTTPS, autenticación y autorización. Existen SDKs disponibles en Python, JavaScript, Java y C# para facilitar su adopción rápida. Aunque es relativamente nuevo, A2A permite a los equipos construir agentes específicos por dominio que pueden colaborar para formar flujos de trabajo complejos, lo que lo convierte en una opción sólida para este tipo de escenarios.

44. Amazon S3 Vectors

Evaluar

Amazon S3 Vectors amplía el servicio de almacenamiento de objetos S3 con capacidades nativas de vectores, ofreciendo almacenamiento de vectores integrado y funcionalidad de búsqueda por similitud. Se integra de forma fluida con el ecosistema de AWS, incluyendo Amazon Bedrock y

OpenSearch, y proporciona funciones adicionales como filtrado por metadatos y gobernanza a través de IAM. Aunque aún es una versión preliminar y está sujeta a restricciones y limitaciones consideramos que su propuesta de valor es convincente. Este enfoque rentable y accesible para el almacenamiento de vectores podría habilitar una variedad de aplicaciones que manejan grandes volúmenes de datos y en las que la baja latencia no es la principal prioridad.

45. Ardoq

Evaluar

Ardoq es una plataforma de arquitectura empresarial (EA) que permite a las organizaciones construir, gestionar y escalar sus bases de conocimiento de arquitectura para planificar de manera más efectiva el futuro. A diferencia de la documentación estática tradicional, propensa a desactualizarse y generar silos, el enfoque basado en datos de Ardoq extrae información de los sistemas existentes para crear un grafo de conocimiento dinámico que se mantiene actualizado a medida que el entorno evoluciona. Una funcionalidad que hemos encontrado especialmente útil es Ardoq Scenarios, que permite modelar y definir visualmente escenarios hipotéticos (what-if) utilizando un enfoque de ramificación y fusión similar a Git. Las organizaciones que buscan una transformación arquitectónica deberían evaluar plataformas de EA dedicadas como Ardoq por su potencial para optimizar y acelerar este proceso.

46. CloudNativePG

Evaluar

CloudNativePG es un Kubernetes Operator que simplifica el alojamiento y la gestión de clústeres de PostgreSQL de alta disponibilidad en Kubernetes. Ejecutar un servicio con estado como PostgreSQL en Kubernetes puede ser complejo y requiere un conocimiento profundo tanto de Kubernetes como de la replicación de PostgreSQL. CloudNativePG abstrae gran parte de esta complejidad al tratar todo el clúster de PostgreSQL como un único recurso declarativo configurable. Proporciona una arquitectura primaria/secundaria sin interrupciones mediante replicación nativa en streaming e incluye funciones de alta disponibilidad integradas, como capacidades de autorreparación, conmutación por error automatizada que promueve la réplica más actualizada y recreación automática de réplicas fallidas. Si estás buscando alojar PostgreSQL en Kubernetes, CloudNativePG es un excelente punto de partida.

47. Coder

Evaluar

Coder es una plataforma para aprovisionar rápidamente entornos de desarrollo estandarizados, siguiendo la práctica de ambientes de desarrollo en la nube que hemos descrito anteriormente. En comparación con herramientas similares como Gitpod (ahora renombrada a Ona) y GitHub Codespaces, Coder ofrece un mayor control sobre la personalización de las estaciones de trabajo mediante Terraform. Aloja las estaciones de trabajo en infraestructura propia, ya sea en la nube o en un centro de datos, en lugar de en los servidores de un proveedor. Este enfoque proporciona más flexibilidad, incluyendo la capacidad de ejecutar agentes de codificación con IA y acceder a sistemas internos de la organización. Sin embargo, esta flexibilidad implica compromisos: mayor esfuerzo para configurar y mantener las plantillas de estaciones de trabajo, y mayor responsabilidad en la gestión de los riesgos de seguridad de los datos en flujos de trabajo agénticos.

48. Graft

Evaluar

Graft es un motor de almacenamiento transaccional que permite una sincronización de datos sólida y eficiente en entornos edge y distribuidos. Lo logra utilizando replicación diferida para sincronizar los datos sólo bajo demanda, replicación parcial para minimizar el consumo de ancho de banda y aislamiento de instantáneas serializables para garantizar la integridad de los datos. Hemos mencionado Electric en el Radar para casos de uso similares, pero consideramos que Graft es único al convertir el almacenamiento de objetos en un sistema transaccional que admite actualizaciones consistentes a nivel de página sin imponer un formato de datos específico. Esto lo hace especialmente adecuado para potenciar aplicaciones móviles local-first mobile applications, managing complex cross-platform synchronization and serving as the backbone for stateless replicas in serverless or embedded systems.

49. groundcover

Evaluar

groundcover es una plataforma de observabilidad cloud-native que unifica logs, trazas, métricas y eventos de Kubernetes en una vista centralizada. Utiliza eBPF para capturar datos de observabilidad granulares sin necesidad de instrumentación en el código, es decir, sin insertar agentes ni SDKs en el código de la aplicación. El sensor eBPF de groundcover se ejecuta en un nodo dedicado dentro de cada clúster monitoreado, funcionando de manera independiente a las aplicaciones que observa. Entre sus principales características se incluyen una visibilidad profunda a nivel del kernel, una arquitectura bring-your-own-cloud (BYOC) architecture que protege la privacidad de los datos y un modelo de precios independiente del volumen de datos, lo que mantiene los costos predecibles.

50. Karmada

Evaluar

Karmada (("Kubernetes Armada") es una plataforma para orquestar cargas de trabajo a través de múltiples clústeres de Kubernetes, nubes y centros de datos. Actualmente, muchos equipos realizan despliegues entre clústeres utilizando herramientas de GitOps como Flux o ArgoCD combinadas con scripts personalizados, por lo que una solución diseñada específicamente para este propósito es bienvenida. Karmada aprovecha las APIs nativas de Kubernetes, sin requerir cambios en las aplicaciones ya construidas para entornos nativos de nube. Ofrece capacidades avanzadas de programación para la gestión multi-nube, alta disponibilidad, recuperación ante fallos y gestión del balanceo de tráfico.

Karmada sigue siendo relativamente nueva, por lo que es importante evaluar la madurez de las funcionalidades de las que dependa tu equipo. Sin embargo, como proyecto de CNCF, tiene un fuerte impulso y varios de nuestros equipos ya lo están utilizando con éxito. Cabe destacar que ciertas áreas como la gestión de redes, estado y almacenamiento entre clústeres están fuera del alcance de Karmada. La mayoría de los equipos aún necesitarán una malla de servicios como Istio o Linkerd fpara el manejo del tráfico y deberán planificar cómo gestionar las cargas de trabajo con estado y los datos distribuidos.

51. OpenFeature

Evaluar

A medida que las empresas escalan, la gestión de feature flags suele volverse cada vez más compleja; los equipos necesitan una capa de abstracción que vaya más allá del feature toggle más simple posible. OpenFeature proporciona esta capa mediante una especificación de API impulsada por la comunidad y agnóstica al proveedor, que estandariza la forma en que se definen y consumen las feature flags, desacoplando el código de la aplicación de la solución de gestión. Esta flexibilidad permite a los equipos cambiar fácilmente de proveedor, desde configuraciones básicas con variables de entorno o configuraciones en memoria, hasta plataformas más avanzadas como ConfigCat o LaunchDarkly. Sin embargo, persiste una advertencia clave: los equipos deben gestionar las diferentes categorías de flags de forma separada y disciplinada para evitar la proliferación de flags, la complejidad de las aplicaciones y la sobrecarga en las pruebas.

52. Oxide

Evaluar

Construir y operar infraestructura privada es una tarea compleja. Esa es una de las principales razones por las que la nube pública se ha convertido en la opción predeterminada para la mayoría de las organizaciones. Sin embargo, para aquellos que la necesitan, Oxide ofrece una alternativa para ensamblar e integrar hardware y software desde cero. Proporciona racks preconstruidos con cómputo, red y almacenamiento, ejecutando software de sistema completamente integrado. Los equipos pueden administrar los recursos a través del API IaaS de Oxide utilizando Terraform y otras herramientas de automatización, en lo que Oxide denomina infraestructura elástica on-premises.

VxRail, Nutanix y HPE SimpliVity también ofrecen soluciones de infraestructura hiperconvergente (HCI), pero lo que distingue a Oxide es su enfoque diseñado específicamente para este propósito. Oxide diseña toda la pila —desde las placas de circuito y las fuentes de alimentación hasta el firmware en lugar de ensamblar componentes de distintos proveedores. Además, ha desarrollado y publicado como código abierto Hubris, un kernel liviano, con protección de memoria y basado en paso de mensajes, escrito en Rust para sistemas embebidos, junto con otros proyectos de infraestructura también basados en Rust. Valoramos además que Oxide venda su equipamiento y software sin tarifas de licencia.

53. Restate

Evaluar

Restate es una plataforma de ejecución duradera diseñada para abordar los desafíos de los sistemas distribuidos complejos al desarrollar aplicaciones con estado y tolerantes a fallos. Registra cada paso mediante un sistema de registro (journaling) de ejecución, lo que garantiza tolerancia a fallos, recuperación confiable y comunicación exactamente una vez (exactly-once) entre servicios. La principal ventaja arquitectónica de la plataforma radica en separar la lógica de aplicación en tres tipos de servicios duraderos: Basic Services, para funciones sin estado; Virtual Objects, para modelar entidades concurrentes con estado; y Workflows, para orquestar procesos complejos y de múltiples pasos. Hemos estado evaluando Restate cuidadosamente en un gran sistema de seguros y, hasta ahora, estamos muy satisfechos con su rendimiento.

54. SkyPilot

Evaluar

SkyPilot es una plataforma de código abierto para ejecutar y escalar cargas de trabajo de IA en entornos locales o en la nube. Desarrollada por el Sky Computing Lab de la Universidad de California en Berkeley, SkyPilot actúa como un intermediario inteligente que busca y aprovisiona automáticamente las GPU más económicas y más disponibles entre los principales proveedores de nube y clústeres de Kubernetes, reduciendo con frecuencia los costos de cómputo. Para los equipos de infraestructura, simplifica la ejecución de IA en Kubernetes al ofrecer la facilidad de uso de Slurm la solidez nativa de la nube, acceso directo por SSH a los pods y características como planificación en grupo y compatibilidad con múltiples clústeres para escalar sin problemas cargas de trabajo de entrenamiento o inferencia.

55. StarRocks

Evaluar

StarRocks es una base de datos analítica que redefine la inteligencia empresarial en tiempo real al combinar la velocidad de los sistemas OLAP tradicionales con la flexibilidad de un data lakehouse. moderno. Alcanza una latencia de consulta de menos de un segundo a gran escala mediante un motor de ejecución optimizado con SIMD, almacenamiento columnar y un sofisticado optimizador basado en costos. Esta arquitectura de alto rendimiento permite ejecutar análisis complejos directamente sobre formatos de datos abiertos como Apache Iceberg, sin necesidad de precomputación ni duplicación de datos. Aunque existen muchas plataformas en este ámbito, consideramos que StarRocks es una opción sólida y rentable para soluciones que requieren alta concurrencia extrema y datos consistentes y actualizados al segundo.

56. Uncloud

Evaluar

Uncloud es una herramienta ligera de orquestación y agrupamiento de contenedores que permite a las personas desarrolladoras llevar aplicaciones de Docker Compose a producción, ofreciendo una experiencia sencilla similar a la de la nube, pero sin la carga operativa de Kubernetes. Logra escalado entre máquinas y despliegues sin tiempo de inactividad al configurar automáticamente una red de malla segura con WireGuard para la comunicación y utilizar el proxy inverso Caddy para proporcionar HTTPS automático y balanceo de carga. La principal ventaja arquitectónica de Uncloud es su diseño completamente descentralizado, que elimina la necesidad de un plano de control central y garantiza que las operaciones del clúster sigan siendo funcionales incluso si algunas máquinas quedan fuera de línea. Con Uncloud, puedes combinar libremente máquinas virtuales en la nube y servidores bare-metal en un entorno informático unificado y rentable.

Herramientas

Adoptar

57. ClickHouse
58. NeMo Guardrails
59. pnpm
60. Pydantic

Probar

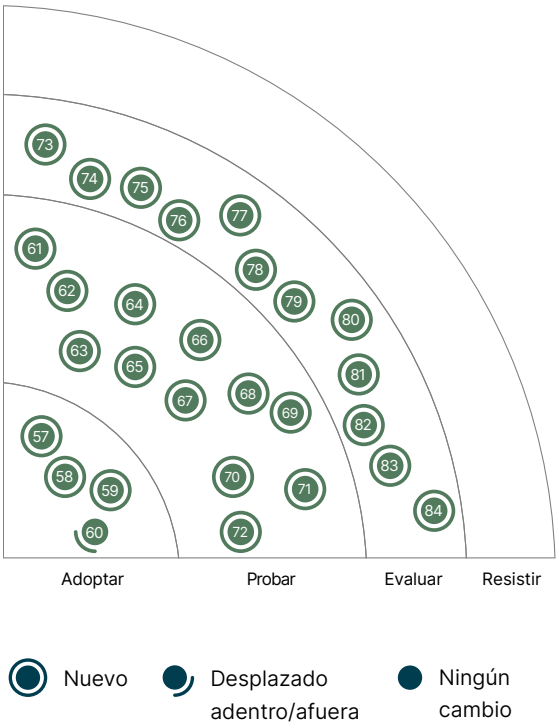
61. AI Design Reviewer
62. Barman
63. Claude Code
64. Cleanlab
65. Context7
66. Data Contract CLI
67. Databricks Assistant
68. Hoppscotch
69. NVIDIA DCGM Exporter
70. RelationalAI
71. UX Pilot
72. v0

Evaluar

73. Augment Code
74. Azure AI Document Intelligence
75. Docling
76. E2B
77. Editor Helix
78. Kueue
79. MCPScan.ai
80. oRPC
81. Power User para dbt
82. Serena
83. SweetPad
84. Tape/Z (Tools for Assembly Program Exploration for Z/OS)

Resistir

—



Nuevo

Desplazado adentro/afuera

Ningún cambio

57. ClickHouse

Adoptar

ClickHouse es una base de datos OLAP distribuida y columnar de código abierto para análisis en tiempo real. Ha madurado hasta convertirse en un motor altamente eficiente y escalable, capaz de manejar análisis de datos a gran escala. Sus vistas materializadas incrementales, su motor de consultas eficiente y su sólida compresión de datos lo hacen ideal para consultas interactivas. La compatibilidad incorporada con funciones agregadas aproximadas permite equilibrar precisión y rendimiento, lo que resulta especialmente útil para análisis de alta cardinalidad. La incorporación del motor de almacenamiento S3 y MergeTree permite la separación del almacenamiento y el cómputo, utilizando almacenamiento compatible con S3 para las tablas de ClickHouse. También hemos encontrado que ClickHouse es un excelente backend para datos de OpenTelemetry y herramientas de análisis de fallos como Sentry. Para los equipos que buscan un motor de análisis rápido y de código abierto, ClickHouse es una excelente opción.

58. NeMo Guardrails

Adoptar

NeMo Guardrails es un kit de herramientas de código abierto desarrollado por NVIDIA que facilita la incorporación de mecanismos programables de seguridad y control en aplicaciones conversacionales basadas en LLMs. Garantiza que las respuestas se mantengan seguras, relevantes y en cumplimiento mediante la definición y aplicación de reglas de comportamiento. Los desarrolladores utilizan Colang, un lenguaje creado específicamente para este propósito, con el fin de crear flujos de diálogo flexibles y gestionar conversaciones aplicando rutas predefinidas y procedimientos operativos. NeMo Guardrails también ofrece una API asíncrona para un mejor rendimiento y soporta salvaguardas para la protección del contenido, seguridad y moderación de entradas y salidas. Estamos observando una adopción constante entre los equipos que construyen aplicaciones que van desde chatbots simples hasta flujos de trabajo complejos basados en agentes. Con su conjunto de funcionalidades en expansión y su cobertura cada vez más madura de vulnerabilidades comunes en LLMs, estamos moviendo NeMo Guardrails a Adopt.

59. pnpm

Adoptar

Desde el último Radar Tecnológico, hemos seguido recibiendo comentarios positivos de los equipos sobre pnpm pnpm es un gestor de paquetes de Node.js que ofrece mejoras significativas de rendimiento en comparación con otras alternativas, tanto en velocidad como en eficiencia del uso del espacio en disco. Utiliza hard links para vincular los paquetes duplicados de las carpetas `node_modules` de múltiples proyectos hacia una única ubicación en el disco y admite optimizaciones incrementales a nivel de archivo que mejoran aún más el rendimiento. Dado que pnpm proporciona un ciclo de retroalimentación mucho más rápido y con problemas mínimos de compatibilidad, se ha convertido en nuestra opción predeterminada para la gestión de paquetes en Node.js.

60. Pydantic

Adoptar

Pydantic es una biblioteca de Python que utiliza las anotaciones de tipo estándar para definir modelos de datos y aplicar esquemas de datos en tiempo de ejecución. Originalmente, las anotaciones de tipo se añadieron a Python para análisis estático, pero su creciente versatilidad ha llevado a usos más amplios, incluida la validación en tiempo de ejecución. Construido sobre un núcleo rápido en Rust, ofrece validación, análisis y serialización de datos eficiente.

Aunque es más conocida por su uso en el desarrollo de APIs web, Pydantic también se ha vuelto esencial en aplicaciones con LLMs. Normalmente utilizamos la técnica de salida estructurada de LLMs para gestionar la naturaleza impredecible de los LLMs. Al definir un esquema de datos estricto, actúa como una red de seguridad frente a la naturaleza impredecible de la salida del modelo, convirtiendo respuestas de texto libre en objetos de Python deterministas y con tipos seguros (por ejemplo, JSON). Este enfoque, a menudo implementado mediante Pydantic AI o LangChain, convierte interacciones potencialmente frágiles con LLMs en contratos de datos confiables y legibles por máquina. Nuestros equipos han utilizado Pydantic con éxito en producción para extraer representaciones estructuradas de documentos no estructurados, asegurando que la salida cumpla con una estructura válida. Por su madurez, rendimiento y confiabilidad, Pydantic es ahora nuestra opción predeterminada para aplicaciones de IA en Python a nivel de producción.

61. AI Design Reviewer

Probar

AI Design Reviewer es un complemento de Figma para realizar auditorías de diseño o evaluaciones heurísticas y recopilar comentarios accionables sobre diseños existentes o nuevos. Sus auditorías abarcan críticas de UX, inconsistencias de UI, brechas de accesibilidad, calidad del contenido y escenarios límite. Más allá de identificar problemas, proporciona recomendaciones con conocimiento del dominio que ayudan a los equipos a construir un vocabulario de diseño compartido y una justificación detrás de las decisiones de diseño. Nuestros equipos utilizaron AI Design Reviewer para analizar diseños heredados, identificando experiencias positivas que debían conservarse y negativas que debían abordarse, lo que sirvió para definir los objetivos de UX en los rediseños. También ha funcionado como un sustituto de la revisión entre pares, ofreciendo feedback temprano sobre nuevos diseños antes de la entrega a desarrollo.

62. Barman

Probar

Barman (Backup and Recovery Manager) es una herramienta de código abierto para gestionar copias de seguridad y recuperación ante desastres en servidores PostgreSQL. Admite el proceso completo de recuperación ante desastres, simplificando la creación de copias de seguridad físicas mediante una variedad de métodos, organizándolas en un catálogo integral y restaurándolas en un servidor activo con capacidades de recuperación a un punto en el tiempo (Point-In-Time Recovery - PITR). Hemos comprobado que Barman es potente y fácil de usar, y nos ha impresionado la velocidad de las operaciones PTIR durante actividades de migración. También hemos comprobado que es eficaz para copias de seguridad programadas, con la capacidad de manejar configuraciones complejas y mixtas de programación y retención.

63. Claude Code

Probar

Claude Code de Anthropic es una herramienta de codificación con IA agéntica que ofrece una interfaz en lenguaje natural y un modelo de ejecución basado en agentes para planificar e implementar flujos de trabajo complejos y de múltiples pasos. Lanzada hace menos de un año, ya ha sido ampliamente adoptada por desarrolladores dentro y fuera de Thoughtworks, motivo por el cual la ubicamos en la categoría Trial. Se han lanzado agentes de codificación basados en consola como Codex CLI, de OpenAI, Gemini CLI de Google y el proyecto de código abierto OpenCode mientras que los asistentes

basados en IDE como Cursor, Windsurf y GitHub Copilot ahora incluyen modos agénticos. Aun así, Claude Code sigue siendo una de las opciones favoritas. Hemos visto equipos que lo utilizan no solo para escribir y modificar código, sino también como un agente de IA de propósito general para gestionar especificaciones, historias, configuración, infraestructura y documentación.

La codificación agéntica cambia el enfoque de las personas desarrolladoras: de escribir código a especificar la intención y delegar la implementación. Aunque esto puede acelerar los ciclos de desarrollo, también puede conducir a una complacencia con el código generado por IA, lo que a su vez puede resultar en código más difícil de mantener y evolucionar, tanto para humanos como para agentes de IA. Por ello, es fundamental que los equipos gestionen rigurosamente cómo se utiliza Claude Code, aplicando técnicas como ingeniería de contexto, prompts compartidos seleccionados y, potencialmente, equipos de agentes de codificación.

64. Cleanlab

Probar

En el paradigma de IA centrada en los datos, mejorar la calidad del conjunto de datos suele generar mayores incrementos de rendimiento que ajustar el modelo en sí. Cleanlab es una biblioteca de Python de código abierto diseñada para abordar este desafío al identificar automáticamente problemas comunes en los datos, como etiquetado incorrecto, valores atípicos y duplicados, en conjuntos de datos de texto, imagen, tabla y audio. Basado en el principio de confident learning, Cleanlab utiliza las probabilidades predichas por el modelo para estimar el ruido en las etiquetas y cuantificar la calidad de los datos.

Este enfoque independiente del modelo permite a las personas desarrolladoras diagnosticar y corregir errores en los conjuntos de datos, y luego volver a entrenar los modelos para mejorar su solidez y precisión. Nuestros equipos han utilizado Cleanlab con éxito en entornos de producción, confirmando su efectividad en escenarios reales. Lo recomendamos como una herramienta valiosa para promover la estandarización de datos y mejorar la calidad de los conjuntos de datos en proyectos de ingeniería de IA.

65. Context7

Probar

Context7 es un servidor MCP que aborda las imprecisiones en el código generado por IA. Mientras los LLMs dependen de datos de entrenamiento desactualizados, Context7 garantiza que generen código preciso, actualizado y específico para la versión de las bibliotecas y frameworks utilizados en un proyecto. Lo consigue extrayendo la documentación más reciente y ejemplos de código funcional directamente de los repositorios de código fuente de los frameworks e inyectándolos en la ventana de contexto del LLM en el momento del prompt. En nuestra experiencia, Context7 ha reducido significativamente las alucinaciones de código y la dependencia de datos de entrenamiento obsoletos. Puede configurarse con editores de código con IA como Claude Code, Cursor o VS Code para generar, refactorizar o depurar código dependiente de frameworks.

66. Data Contract CLI

Probar

Data Contract CLI es una herramienta de línea de comandos de código abierto diseñada para trabajar con la especificación de Data Contract Ayuda a crear y editar contratos de datos y, de forma crucial, permite validar los datos frente a su contrato, lo cual es esencial para garantizar la integridad y la calidad de los productos de datos.

La CLI ofrece un amplio soporte para múltiples definiciones de esquema (Avro, SQL DDL, Open Data Contract Standard, entre otros) y puede comparar diferentes versiones de contratos para detectar de inmediato cambios incompatibles. La hemos encontrado especialmente útil en el ámbito de data mesh para operacionalizar la gobernanza de contratos entre productos de datos mediante la integración en CI/CD. Este enfoque reduce los errores manuales y garantiza la calidad, integridad y compatibilidad de los datos en los intercambios entre servicios.

67. Databricks Assistant

Probar

Databricks Assistant es una herramienta conversacional impulsada por IA integrada directamente en la plataforma Databricks, que actúa como un programador en pareja contextual para profesionales de datos. A diferencia de los asistentes de código de propósito general, se beneficia de una comprensión nativa del entorno y del contexto de datos de Databricks, incluido los metadatos de Unity Catalog. Assistant va más allá de generar fragmentos de código: puede crear consultas SQL y Python complejas de múltiples pasos, diagnosticar errores y ofrecer explicaciones detalladas y específicas del entorno de trabajo. Para las organizaciones que ya utilizan el ecosistema de Databricks, puede acelerar la productividad y reducir la barrera de entrada a tareas de datos complejas.

68. Hoppscotch

Probar

Hoppscotch es una herramienta ligera y de código abierto para el desarrollo, depuración, prueba y compartición de APIs. Admite múltiples protocolos, incluidos HTTP, GraphQL y WebSocket y ofrece clientes multiplataforma para entornos web, de escritorio y CLI.

Aunque el espacio de herramientas para APIs está lleno de alternativas como Postman, Insomnia y Bruno, Hoppscotch destaca por su bajo consumo de recursos y su diseño orientado a la privacidad. No incluye analíticas, utiliza almacenamiento local-first y admite implementación autogestionada (self-hosting). Es una opción sólida para las organizaciones que buscan una forma intuitiva de compartir scripts de API mientras mantienen una fuerte protección de los datos.

69. NVIDIA DCGM Exporter

Probar

NVIDIA DCGM Exporter es una herramienta de código abierto que ayuda a los equipos a monitorizar el entrenamiento distribuido en GPU a gran escala. Convierte la telemetría propietaria del NVIDIA Data Center GPU Manager (DCGM) en formatos abiertos compatibles con sistemas de monitorización estándar. El Exporter expone métricas críticas en tiempo real, incluidas la utilización de GPU, temperatura, consumo de energía y conteos de errores ECC, tanto de las GPU como de los servidores anfitriones. Esta visibilidad es esencial para las organizaciones que ajustan modelos LLM

personalizados o ejecutan trabajos de entrenamiento prolongados e intensivos en GPU. El efecto straggler, donde un solo trabajador (worker) lento genera un cuello de botella en todo el proceso, puede reducir el rendimiento en más de un 10% y desperdiciar hasta un 45% de las horas de GPU asignadas. Diseñado para entornos cloud-native y de gran escala, DCGM Exporter se integra sin problemas con Prometheus y Grafana, ayudando a garantizar que cada GPU opere dentro de los límites óptimos de rendimiento.

70. RelationalAI

Probar

Cuando se incorporan grandes volúmenes de datos diversos en Snowflake, las relaciones inherentes y las reglas implícitas dentro de esos datos pueden volverse difíciles de identificar. Desarrollado como una aplicación nativa de Snowflake, RelationalAI permite a los equipos crear modelos sofisticados que capturan conceptos significativos, definen entidades de negocio centrales e integran lógica compleja directamente sobre las tablas de Snowflake. Su potente Graph Reasoner permite crear, analizar y visualizar grafos de conocimiento relacionales basados en estos modelos. Los algoritmos integrados ayudan a explorar estructuras de grafo y revelar patrones ocultos. Para las organizaciones que gestionan conjuntos de datos masivos y en constante cambio, construir un grafo de conocimiento puede ser fundamental para el monitoreo proactivo y la generación de información más rica y accionable.

71. UX Pilot

Probar

UX Pilot es una herramienta de IA que soporta múltiples etapas del proceso de diseño UX, desde la creación de wireframes hasta el diseño visual de alta fidelidad y la revisión. Acepta entradas de texto o imagen y puede generar automáticamente pantallas, flujos y diseños. Su función Autoflow crea transiciones de flujo de usuario, mientras que Deep Design produce resultados más ricos y detallados. UX Pilot también incluye un complemento de Figma que exporta los diseños generados para su refinamiento dentro de las herramientas de diseño estándar. Nuestros equipos han utilizado UX Pilot para la ideación e inspiración, generando múltiples opciones durante los ejercicios de Crazy 8 y traduciendo listas de historias de proyecto en tableros de visión de producto y conceptos de diseño a nivel épico. Herramientas como UX Pilot también permiten que personas no diseñadoras, como las product managers, creen prototipos rápidos y obtengan retroalimentación temprana de las partes interesadas, una tendencia en crecimiento dentro de los flujos de trabajo de diseño asistido por IA.

72. v0

Probar

v0 ha evolucionado desde la última vez que apareció en el Radar. Ahora incluye un modo de diseño que reduce aún más la barrera para que los gerentes de producto creen y ajusten prototipos de interfaz de usuario de autoservicio. La versión más reciente incorpora un modelo propio con amplias ventanas de contexto y capacidades multimodales, lo que permite a v0 generar y mejorar interfaces de usuario a partir de entradas tanto de texto como visuales. Otra incorporación destacada es su modo agente, que permite al sistema descomponer trabajos más complejos y seleccionar el modelo apropiado para cada caso. Sin embargo, esta función aún es nueva y los primeros comentarios han sido mixtos.

73. Augment Code

Evaluar

Augment Code es un asistente de codificación con IA que ofrece soporte profundo y contextual en grandes bases de código. Se destaca por su avanzada ingeniería de contexto que permite actualizaciones rápidas del índice de código y una recuperación eficiente, incluso cuando el código cambia con frecuencia. Augment es compatible con modelos como Claude Sonnet 4 and 4.5 y GPT-5, se integra con GitHub, Jira y Confluence, y admite el Model Context Protocol (MCP) para la interoperabilidad con herramientas externas. Ofrece orientación paso a paso para realizar cambios complejos en bases de código, desde refactorizaciones y actualizaciones de dependencias hasta modificaciones de esquemas, junto con autocompletados personalizados en línea que reflejan las dependencias específicas de cada proyecto. Además, Augment fomenta la colaboración al permitir que los equipos consulten y compartan información del código directamente dentro de Slack.

74. Azure AI Document Intelligence

Evaluar

Azure AI Document Intelligence (anteriormente Form Recognizer) extrae texto, tablas y pares clave-valor de documentos no estructurados y los transforma en datos estructurados. Utiliza modelos de deep learning preentrenados para interpretar el diseño y la semántica, y permite entrenar modelos personalizados mediante una interfaz sin código para formatos específicos. Sin embargo, en algunos casos, las personas usuarias avanzadas pueden necesitar una interfaz de ajuste fino personalizada.

Uno de nuestros equipos informó que ADI redujo significativamente la entrada manual de datos, mejoró la precisión y aceleró la generación de reportes, lo que permitió tomar decisiones basadas en datos con mayor rapidez. Al igual que Amazon Textract y Google Document AI, ofrece procesamiento de documentos a nivel empresarial con una sólida comprensión del diseño. Una alternativa de código abierto emergente es Docling de IBM, que ofrece un enfoque más flexible y centrado en el código para la extracción de datos estructurados. En comparación con las herramientas tradicionales de OCR, ADI no solo captura texto, sino también estructura y relaciones, lo que facilita su integración en pipelines de datos posteriores. Dicho esto, hemos observado cierta latencia al integrarlo en flujos de trabajo síncronos, por lo que recomendamos usarlo principalmente para procesamiento asíncrono.

75. Docling

Evaluar

Docling es una biblioteca de código abierto de Python y TypeScript para el procesamiento avanzado de documentos con datos no estructurados. Aborda el problema, a menudo pasado por alto, del "último tramo" al convertir documentos del mundo real, como archivos PDF y presentaciones en PowerPoint, en formatos limpios y legibles por máquina. A diferencia de los extractores tradicionales, Docling utiliza un enfoque basado en visión por computadora para interpretar la estructura semántica y el diseño del documento, lo que hace que sus resultados sean especialmente valiosos para los pipelines de generación aumentada por recuperación (RAG) Convierte documentos complejos en formatos estructurados como JSON o Markdown, lo que permite aplicar técnicas como salida estructurada de LLMs. Esto contrasta con ColPali, que envía imágenes de páginas directamente a un modelo visión-lenguaje para su recuperación.

La naturaleza open source de Docling y su núcleo en Python, construido sobre un modelo de datos basado en Pydantic ofrecen una alternativa flexible y autogestionada frente a herramientas

propietarias en la nube como Azure Document Intelligence, Amazon Textract y Google Document AI. Respaldado por IBM Research, el rápido desarrollo del proyecto y su arquitectura lista para integrarse con otros frameworks como LangGraph lo convierten en una opción muy recomendable para los equipos que construyen pipelines de datos listos para IA de nivel productivo.

76. E2B

Evaluar

E2B es una herramienta de código abierto para ejecutar código generado por IA en entornos seguros y aislados (sandboxes) en la nube. Los agentes pueden utilizar estos entornos, construidos sobre microVMs de Firecracker para ejecutar código de forma segura, analizar datos, realizar investigaciones o manejar una máquina virtual. Esto permite crear e implementar agentes de IA de nivel empresarial con control y seguridad total sobre el entorno de ejecución.

77. Editor Helix

Evaluar

Ha habido un cierto resurgimiento de editores de texto simples que buscan reemplazar al favorito de línea de comandos, Vim. Helix es uno de los competidores en este espacio, junto con Neovim y, más recientemente, Kakoune. Describiéndose a sí mismo, de manera algo lúdica, como un editor de texto posmoderno, Helix incluye cursores múltiples, soporte para Tree-sitter y compatibilidad integrada con el Protocolo de Servidor de Lenguaje (LSP), lo que inicialmente llamó nuestra atención. Helix se está desarrollando de forma activa y pronto contará con un sistema de complementos (plugins). En general, es un editor modal liviano que resultará familiar para los usuarios de Vim, pero que incorpora algunas comodidades modernas.

78. Kueue

Evaluar

Kueue es un controlador nativo de Kubernetes para la gestión de colas de trabajos que administra cuotas y consumo de recursos. Proporciona APIs para manejar cargas de trabajo en Kubernetes con diferentes prioridades y requisitos de recursos, funcionando como un gestor a nivel de trabajos (jobs) que determina cuándo admitirlos o expulsarlos. Diseñado para la gestión eficiente de recursos, la priorización de trabajos y la planificación (schedule) avanzada, Kueue ayuda a optimizar la ejecución de cargas de trabajo en entornos Kubernetes, particularmente para cargas de trabajo de ML que utilizan herramientas como Kubeflow. Trabaja junto a cluster-autoscaler y a kube-scheduler en lugar de reemplazarlos, enfocándose en la admisión de trabajos según el orden, la cuota, la prioridad y la consciencia de la topologia (topology awareness). Como parte del ecosistema Kubernetes Special Interest Group (SIG), Kueue sigue sus estándares de desarrollo.

79. MCPScan.ai

Evaluar

MCPScan.ai es un escáner de seguridad para servidores Model Context Protocol (MCP) que opera en dos modos: escaneo y proxy. En el modo de escaneo, analiza las configuraciones y descripciones de herramientas para detectar vulnerabilidades conocidas como inyecciones de prompt, tool poisoning y flujos tóxicos. En el modo proxy, MCPScan.ai actúa como un puente entre el sistema de agentes y el servidor MCP, monitoreando continuamente el tráfico en tiempo de ejecución. Este modo también

aplica reglas y límites de seguridad personalizados, como validación de llamadas a herramientas, detección de información personal (PII) y restricciones en los flujos de datos. La herramienta proporciona una capa proactiva de seguridad para agentes, garantizando que incluso si se acepta un prompt malicioso, el agente no pueda ejecutar acciones perjudiciales. MCPScan.ai es una solución de seguridad diseñada específicamente para el campo emergente de los sistemas basados en agentes.

80. oRPC

Evaluar

oRPC (OpenAPI Remote Procedure Call) ofrece API con seguridad de tipos de extremo a extremo en TypeScript, cumpliendo completamente con la especificación OpenAPI. Puede generar automáticamente una especificación OpenAPI completa, lo que simplifica la integración y la documentación. Hemos encontrado que oRPC es especialmente sólido para integraciones. Mientras que alternativas como tRPC y ElysiaJS suelen requerir adoptar un nuevo framework para obtener tipado seguro, oRPC se integra sin problemas con frameworks existentes de Node.js, Express, Fastify, Hono y Next.js. Esta flexibilidad lo convierte en una excelente opción para los equipos que buscan incorporar seguridad de tipos de extremo a extremo en sus API existentes sin una refactorización disruptiva.

81. Power User para dbt

Evaluar

Power user para dbt es una extensión para Visual Studio Code que se integra directamente con dbt y entornos de dbt en la nube. Dado que dbt sigue siendo una de nuestras herramientas favoritas, cualquier mejora en su usabilidad es una valiosa incorporación al ecosistema. Antes, las personas desarrolladoras dependían de varias herramientas para validar el código SQL o inspeccionar la genealogía de los modelos fuera del IDE. Con esta extensión, esas capacidades ahora están integradas en VS Code, ofreciendo autocompletado de código, resultados de consulta en tiempo real y visualización de la genealogía de modelos y columnas. Esta última función facilita la navegación entre modelos. Nuestros equipos informan que el complemento reduce los errores en los pipelines y mejora la experiencia general de desarrollo. Si utilizas dbt, te recomendamos que le eches un vistazo a esta herramienta.

82. Serena

Evaluar

Serena es un potente conjunto de herramientas de codificación que equipa a los agentes de programación, como Claude Code, con capacidades similares a las de un IDE para la recuperación y edición semántica de código. Al operar a nivel de símbolos y comprender la estructura relacional del código, Serena mejora enormemente la eficiencia en el uso de tokens. En lugar de leer archivos completos o depender de reemplazos de cadenas rudimentarios, los agentes de programación pueden usar herramientas precisas de Serena, como find_symbol, find_referencing_symbols e insert_after_symbol, para localizar y editar el código. Aunque el impacto es mínimo en proyectos pequeños, esta eficiencia resulta extremadamente valiosa a medida que la base de código crece.

83. SweetPad

Evaluar

La extensión SweetPad permite a los desarrolladores usar VS Code o Cursor durante todo el ciclo de desarrollo de aplicaciones Swift en plataformas Apple. Elimina la necesidad de cambiar constantemente a Xcode al integrar herramientas esenciales como xcodebuild, xcode-build-server y swift-format. Los desarrolladores pueden compilar, ejecutar y depurar aplicaciones Swift para iOS, macOS y watchOS directamente desde sus IDE, además de gestionar simuladores y desplegar en dispositivos sin abrir Xcode.

84. Tape/Z (Tools for Assembly Program Exploration for Z/OS)

Evaluar

Tape/Z (Tools for Assembly Program Exploration for Z/OS) es un conjunto de herramientas en evolución para analizar código HLASM (High-Level Assembler) de mainframe. Desarrollado por una persona de Thoughtworks, ofrece funcionalidades como análisis sintáctico, construcción de grafos de flujo de control, rastreo de dependencias y visualización de diagramas de flujo. Desde hace tiempo hemos señalado la escasez de herramientas abiertas y comunitarias en el entorno mainframe, donde la mayoría de las opciones siguen siendo propietarias o dependientes de ecosistemas de proveedores. Tape/Z ayuda a cerrar esa brecha al ofrecer capacidades de análisis accesibles y programables. Junto con COBOL REKT un conjunto de herramientas complementario para COBOL que también hemos utilizado en múltiples ocasiones con clientes, representa un avance alentador hacia un ecosistema de herramientas modernas y más amigables para el desarrollo en sistemas mainframe.

Lenguajes y Frameworks

Adoptar

85. Fastify
86. LangGraph
87. vLLM

Probar

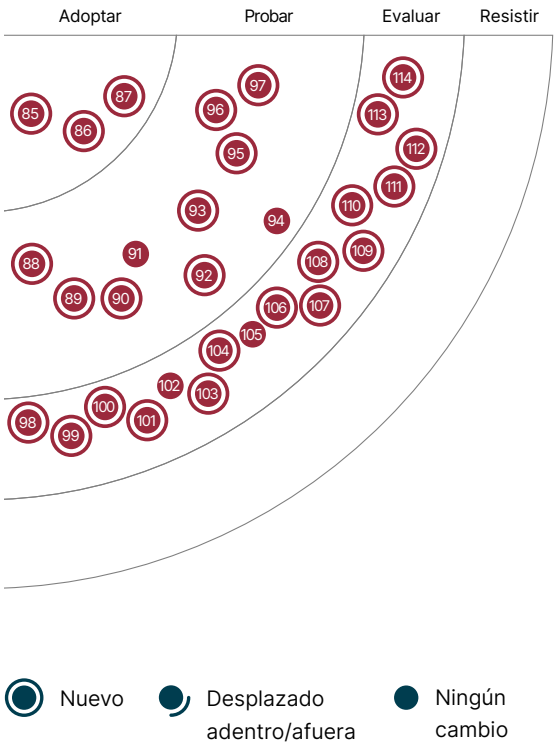
88. Crossplane
89. DeepEval
90. FastMCP
91. LiteLLM
92. MLForecast
93. Nuxt
94. Phoenix
95. Presidio
96. Pydantic AI
97. Tauri

Evaluar

98. Agent Development Kit (ADK)
99. Agno
100. assistant-ui
101. AutoRound
102. Browser Use
103. DeepSpeed
104. Drizzle
105. Criptografía poscuántica en Java
106. kagent
107. LangExtract
108. Langflow
109. LMCache
110. Mem0
111. Open Security Control Assessment Language (OSCAL)
112. OpenInference
113. Valibot
114. Vercel AI SDK

Resistir

—



85. Fastify

Adoptar

Seguimos teniendo una experiencia positiva con Fastify un framework web rápido, no prescriptivo y de bajo consumo de recursos para Node.js. Proporciona todas las capacidades esenciales de un framework web minimalista (incluyendo análisis, validación y serialización), junto con un sólido sistema de complementos y una comunidad activa. Nuestros equipos no han encontrado desventajas significativas al usar Fastify frente a alternativas como Express.js, y además han obtenido mejoras de rendimiento medibles, lo que lo convierte en una opción atractiva para el desarrollo web minimalista en Node.js.

86. LangGraph

Adoptar

LangGraph es un framework de orquestación para crear aplicaciones multi-agente con estado utilizando LLMs. Proporciona primitivas de bajo nivel como nodos y aristas, junto con funciones integradas que ofrecen a las personas desarrolladoras un control granular sobre los flujos de trabajo de los agentes, la gestión de memoria y la persistencia del estado. Esto significa que las personas desarrolladoras pueden comenzar con un grafo preconstruido sencillo y escalar hacia arquitecturas de agentes complejas en evolución. Con soporte para streaming, gestión avanzada de contexto y patrones de resiliencia como “model fallbacks” y manejo de errores de herramientas, LangGraph permite crear aplicaciones basadas en agentesagénticas robustas y listas para producción. Su enfoque basado en grafos garantiza flujos de trabajo predecibles y personalizables, y simplifica la depuración y el escalado. Nuestros equipos han obtenido buenos resultados utilizando LangGraph para construir sistemas multiagente gracias a su diseño modular y ligero.

87. vLLM

Adoptar

vLLM es un motor de inferencia de alto rendimiento y de uso eficiente de memoria para LLMs, que puede ejecutarse tanto en la nube como en entornos locales. Es compatible con múltiples arquitecturas de modelos y con modelos de código abierto populares. Nuestros equipos implementan workers de vLLM en contenedores Docker sobre plataformas GPU como NVIDIA DGX e Intel HPC, alojando modelos como Llama 3.1 (8B and 70B), Mistral 7B y Llama-SQL fpara asistencia en codificación, búsqueda de conocimiento e interacciones con bases de datos en lenguaje natural. vLLM es compatible con el estándar del SDK de OpenAI, lo que permite una prestación de modelos coherente. El AI Model Catalog de Azure utiliza un contenedor de inferencia personalizado basado en vLLM para mejorar el rendimiento, siendo vLLM el motor de inferencia predeterminado debido a su alto rendimiento y gestión eficiente de memoria. El framework vLLM se ha convertido en la opción preferida para implementaciones de modelos a gran escala.

88. Crossplane

Probar

Desde su última aparición en el Radar, la adopción de Crossplane ha seguido creciendo, especialmente para extender los clústeres de Kubernetes. En nuestra experiencia, Crossplane destaca en casos de uso específicos más que como una herramienta general de infraestructura como código (IaC). Nuestras observaciones anteriores siguen siendo válidas: Crossplane funciona mejor como complemento de las cargas de trabajo desplegadas dentro de Kubernetes, no como un reemplazo completo de herramientas como Terraform. Los equipos que apostaron por usar Crossplane como su solución principal de IaC a menudo enfrentaron dificultades, mientras que aquellos que lo emplearon de manera pragmática, en casos personalizados y puntuales, obtuvieron buenos resultados.

Delegar la gestión del ciclo de vida de los recursos a Crossplane y utilizar las API de XRD para personalizaciones ligeras ha resultado especialmente eficaz. Crossplane es particularmente valioso para gestionar recursos cuyo ciclo de vida es simple pero no nativo de Kubernetes. Si bien ahora puede crear clústeres de Kubernetes, una capacidad que antes no tenía, recomendamos precaución al adoptarlo como sustituto total de Terraform. En nuestra experiencia, Crossplane ofrece mejores resultados cuando IaC actúa como capa base y Crossplane se utiliza encima para cubrir necesidades especializadas.

89. DeepEval

Probar

DeepEval es un framework de evaluación de código abierto basado en Python para medir el rendimiento de los LLMs. Puede utilizarse para evaluar la generación mejorada por recuperación (RAG) y otras aplicaciones desarrolladas con frameworks como LlamaIndex o LangChain, así como para establecer líneas base y realizar comparaciones de modelos. DeepEval va más allá de las métricas basadas en coincidencia de palabras, evaluando precisión, relevancia y consistencia para ofrecer resultados de evaluación más confiables en escenarios reales. Incluye métricas como detección de alucinaciones, relevancia de respuestas y optimización de hiperparámetros, y admite GEval for creating custom, use case–specific metrics. Our teams are using DeepEval to fine-tune agentic outputs using the LLM como juez Se integra con pytest y con pipelines de CI/CD, lo que facilita su adopción y lo hace valioso para la evaluación continua. Para los equipos que desarrollan aplicaciones basadas en LLM en entornos regulados, Inspect AI, desarrollado por el UK AI Safety Institute, ofrece una alternativa con un enfoque más sólido en auditoría y cumplimiento.

90. FastMCP

Probar

El Model Context Protocol (MCP) se está convirtiendo rápidamente en un estándar para proporcionar contexto y herramientas a las aplicaciones basadas en LLM. Sin embargo, implementar un servidor MCP normalmente implica una cantidad considerable de plantillas de configuración, manejo de protocolos y gestión de errores. FastMCP es un framework de Python que simplifica este proceso al abstraer la complejidad del protocolo y permitir que las personas desarrolladoras definan recursos y herramientas MCP mediante decoradores de Python intuitivos. Esta abstracción permite que los equipos se concentren en la lógica de negocio, lo que da como resultado implementaciones MCP más limpias y fáciles de mantener.

Aunque FastMCP 1.0 ya está incorporado en el official SDK, el estándar MCP continúa evolucionando rápidamente. Recomendamos seguir de cerca la versión 2.0 y asegurarse de que los equipos se mantengan alineados con los cambios en la especificación oficial.

91. LiteLLM

Probar

LiteLLM es un SDK que permite una integración fluida con múltiples proveedores de LLMs a través de un formato estandarizado de API de OpenAI. Admite una amplia variedad de proveedores y modelos, ofreciendo una interfaz unificada para tareas de completado de texto, embeddings y generación de imágenes. Al abstraer las diferencias específicas entre las APIs de los distintos proveedores, LiteLLM simplifica la integración y enruta automáticamente las solicitudes al endpoint del modelo correspondiente. También incluye funciones de nivel empresarial como salvaguardas y mecanismos de control, almacenamiento en caché, registro de logs, limitación de velocidad y balanceo de carga

mediante su proxy framework. A medida que las organizaciones integran aplicaciones impulsadas por IA de manera más profunda en sus flujos de trabajo, la gobernanza y la observabilidad se vuelven esenciales. Nuestros equipos han estado utilizando LiteLLM como un AI gateway para estandarizar, proteger y obtener visibilidad del uso de IA en toda la organización.

92. MLForecast

Probar

MLForecast es un framework y librería de Python para pronóstico de series temporales que aplica modelos de machine learning a conjuntos de datos a gran escala. Simplifica el proceso habitualmente complejo de ingeniería automática de características —incluyendo retrasos, estadísticas móviles y variables basadas en fechas— y es una de las pocas bibliotecas con soporte nativo para frameworks de computación distribuida como Spark y Dask, garantizando escalabilidad. También admite pronósticos probabilísticos mediante métodos como la predicción conformal, lo que proporciona medidas cuantitativas de incertidumbre en las proyecciones. En nuestra evaluación, MLForecast escaló de manera eficiente a millones de puntos de datos y superó de forma consistente a herramientas comparables. Para los equipos que buscan operacionalizar rápidamente el pronóstico de series temporales sobre datos de gran volumen, MLForecast es una opción muy atractiva.

93. Nuxt

Probar

Nuxt es un meta-framework dogmático, construido sobre Vue.js para crear aplicaciones web full-stack, conocido con frecuencia como el "Next.js para Vue.js". Al igual que su contraparte en React, Nuxt ofrece capacidades optimizadas para SEO, como el pre-renderizado, el renderizado en el servidor (Server-Side Rendering - SSR) y la gestión de metadatos, lo que lo convierte en una excelente opción para desarrollar sitios web de alto rendimiento y optimizados para motores de búsqueda sobre la base de Vue.js.

Nuxt cuenta con el respaldo de Vercel, la misma empresa detrás de Next.js, y con una sólida comunidad y un ecosistema de módulos oficiales y de terceros. Estos módulos simplifican la integración de funcionalidades como el procesamiento de imágenes, la generación de mapas del sitio y el uso de Tailwind CSS. Nuxt es una buena opción para los equipos que buscan un framework integral y dogmático para crear aplicaciones optimizadas para SEO con Vue.js.

94. Phoenix

Probar

Seguimos teniendo experiencias positivas con Phoenix, un framework web MVC del lado del servidor escrito en Elixir. Phoenix se basa en los aprendizajes de desarrollo rápido de aplicaciones y experiencia del desarrollador de Ruby on Rails, al mismo tiempo que avanza hacia los paradigmas de la programación funcional.

En esta edición del Radar, destacamos el lanzamiento de Phoenix LiveView 1.0. LiveView es una solución de HTML-over-the-wire —similar a HTMX o Hotwire — que permite a los desarrolladores crear experiencias de usuario enriquecidas y en tiempo real completamente con HTML renderizado en el servidor. Mientras que tecnologías similares suelen manejar actualizaciones parciales mediante

el intercambio de fragmentos HTML, LiveView ofrece una arquitectura completa basada en componentes a través de LiveComponents, con composición, paso de propiedades, gestión de estado y hooks de ciclo de vida similares a React o Vue.js. LiveView combina la productividad y escalabilidad de Phoenix con una sólida gestión de la complejidad, lo que lo hace ideal para construir interfaces altamente interactivas sin necesidad de un framework completo de JavaScript.

95. Presidio

Probar

Presidio es un SDK de protección de datos para identificar y anonimizar información sensible en texto estructurado y no estructurado. Detecta información de identificación personal (PII), como números de tarjetas de crédito, nombres y ubicaciones, utilizando reconocimiento de entidades nombradas, expresiones regulares y lógica basada en reglas. Presidio admite reconocedores de entidades personalizados y pipelines de desidentificación, lo que permite a las organizaciones adaptarlo a sus necesidades de privacidad y cumplimiento normativo. Nuestros equipos han utilizado Presidio en entornos empresariales con controles estrictos sobre el de intercambio de datos al integrarlo con LLMs. Aunque automatiza la detección de información sensible, no es infalible y puede omitir o identificar incorrectamente algunas entidades. Los equipos deben ser cautelosos y actuar con precaución al tomar sus resultados.

96. Pydantic AI

Probar

Pydantic AI continúa demostrando ser un framework de código abierto estable y bien soportado para construir agentes de IA Generativa en entornos de producción. Construido sobre la base confiable de Pydantic ofrece una sólida seguridad de tipado, observabilidad de primer nivel mediante OpenTelemetry y herramientas de evaluación integradas. El lanzamiento de la versión 1.0 el 4 de septiembre de 2025 marcó un hito importante en su madurez. Desde entonces lo hemos encontrado confiable y ampliamente adoptado por su simplicidad y facilidad de mantenimiento, uniéndose a otros frameworks populares para agentes como LangChain y LangGraph.

Las actualizaciones recientes han facilitado la implementación de servidores y clientes Model Context Protocol (MCP), con soporte adicional para estándares emergentes como AG UI y A2A. Con su API limpia y un ecosistema en crecimiento, Pydantic AI se ha convertido en una opción atractiva para nuestros equipos que construyen aplicaciones GenAI listas para producción en Python.

97. Tauri

Probar

Tauri es un framework para crear aplicaciones de escritorio de alto rendimiento utilizando una única base de código de interfaz web. A diferencia de los contenedores web tradicionales como Electron, Tauri está construido sobre Rust y aprovecha el webview nativo del sistema operativo, lo que da como resultado binarios más pequeños y mayor seguridad. Evaluamos Tauri por primera vez hace varios años; desde entonces, se ha expandido más allá del escritorio para ofrecer soporte a iOS y Android. La versión más reciente introduce un modelo de permisos y alcance más flexible, reemplaza la lista de permisos anterior y presenta una capa de comunicación entre procesos (IPC) reforzada que admite transferencia de datos en bruto y mejora el rendimiento. Estas actualizaciones cuentan con el respaldo de una auditoría de seguridad externa. Junto con las directrices oficiales de distribución para las principales tiendas de aplicaciones, estas mejoras fortalecen aún más la posición de Tauri en el ámbito del desarrollo multiplataforma.

98. Agent Development Kit (ADK)

Evaluar

Agent Development Kit (ADK) es un framework para desarrollar y desplegar agentes de IA que aplica principios modernos de ingeniería de software en lugar de depender únicamente del prompting. Introduce abstracciones familiares como clases, métodos, patrones de flujo de trabajo y soporte para CLI. En comparación con frameworks como LangGraph o CrewAI, la fortaleza de ADK radica en su profunda integración con la infraestructura de IA de Google, ofreciendo grounding de nivel empresarial, acceso a datos y monitoreo listos para producción. También está diseñado para la interoperabilidad, con soporte para wrappers de herramientas y el protocolo A2A para comunicación entre agentes. Para las organizaciones que ya usan GCP, ADK representa una base prometedora para construir arquitecturas de agentes escalables, seguras y gestionables. Aunque aún está en sus primeras etapas de evolución, marca la dirección de Google hacia un entorno nativo y completo para el desarrollo de agentes. Recomendamos seguir de cerca su madurez y el crecimiento de su ecosistema.

99. Agno

Evaluar

Agno es un framework para crear, ejecutar y gestionar sistemas multiagente. Ofrece la flexibilidad de desarrollar agentes completamente autónomos o flujos de trabajo controlados por pasos, con soporte integrado para human-in-the-loop, gestión de sesiones, memoria y conocimiento. Valoramos su enfoque en la eficiencia, con tiempos de inicio de agentes impresionantemente rápidos y bajo consumo de memoria. Agno también incluye su propio entorno de ejecución, AgentOS, una aplicación basada en FastAPI con un plano de control integrado que facilita las pruebas, la supervisión y la gestión de sistemas agénticos.

100. assistant-ui

Evaluar

assistant-ui es una biblioteca de código abierto en TypeScript y React para interfaces de chat con IA. Gestiona las partes complejas de la implementación de una interfaz de chat, como el streaming, la gestión de estado para la edición de mensajes y el cambio de ramas, además de funciones comunes de UX, al mismo tiempo que permite a las personas desarrolladoras diseñar sus propios componentes utilizando primitivas de Radix. Admite integración con entornos de ejecución populares, incluidos Vercel AI SDK y LangGraph, y ofrece soluciones de ejecución personalizables para casos de uso complejos. Hemos creado con éxito una interfaz de chat sencilla con assistant-ui y los resultados han sido satisfactorios.

101. AutoRound

Evaluar

AutoRound de Intel es un algoritmo avanzado de cuantización diseñado para comprimir grandes modelos de IA, como LLMs y modelos de lenguaje-visual (VLMs), con una pérdida mínima de precisión. Reduce el tamaño del modelo a anchos de bit ultra bajos (2–4 bits) utilizando optimización por descenso de gradiente de signo y aplica anchos de bit mixtos entre capas para lograr una eficiencia óptima. Este proceso de cuantización también es notablemente rápido: es posible cuantizar un modelo de siete mil millones de parámetros en solo unos minutos con una única GPU. Dado que AutoRound se integra con motores de inferencia populares como vLLM y Transformers, es una opción atractiva para la cuantización de modelos.

102. Browser Use

Evaluar

Browser Use es una biblioteca de Python de código abierto que permite a los agentes basados en LLM operar navegadores web e interactuar con aplicaciones web. Puede navegar, introducir datos, extraer texto y gestionar múltiples pestañas para coordinar acciones en varias aplicaciones. La biblioteca es especialmente útil cuando los agentes de IA necesitan acceder, manipular o recuperar información de contenido web. Es compatible con distintos LLM y utiliza Playwright para combinar la comprensión visual con la extracción de estructura HTML y así lograr interacciones web más completas.

Nuestros equipos integraron Browser Use con el framework Pytest y los informes de Allure para explorar pruebas automatizadas con LLMs. Los pasos de prueba se redactaron en lenguaje natural para que el agente los ejecutara, tomando capturas de pantalla en las aserciones o cuando se producían errores. El objetivo era habilitar QA fuera del horario laboral extrayendo automáticamente los casos de prueba desde Confluence para su verificación posterior al desarrollo. Los resultados iniciales son prometedores, aunque las respuestas del agente tras completar la tarea suelen carecen de descripciones detalladas de los errores, por lo que requiere un sistema de informes de errores personalizado.

103. DeepSpeed

Evaluar

DeepSpeed es una biblioteca de Python que optimiza el aprendizaje profundo distribuido tanto para entrenamiento como para inferencia. Durante el entrenamiento, integra tecnologías como el Zero Redundancy Optimizer (ZeRO) y el paralelismo 3D para escalar modelos de manera eficiente en miles de GPU. En la inferencia, combina paralelismo de tensores, de pipelines, de expertos y ZeRO con núcleos personalizados y optimizaciones de comunicación para reducir la latencia. DeepSpeed ha impulsado algunos de los modelos de lenguaje más grandes del mundo, como Megatron-Turing NLG (530B) y BLOOM (176B). Admite modelos densos y dispersos, ofrece un alto rendimiento del sistema y permite realizar entrenamiento o inferencia en múltiples GPU con recursos limitados. La biblioteca se integra fácilmente con frameworks populares como Hugging Face Transformers, PyTorch Lightning y Accelerate, lo que la convierte en una opción altamente eficaz para cargas de trabajo de aprendizaje profundo a gran escala o con recursos restringidos.

104. Drizzle

Evaluar

Drizzle es un ORM ligero para TypeScript. A diferencia de Prisma ORM, ofrece a los desarrolladores tanto una API simple similar a SQL como una interfaz de consultas más tradicional al estilo ORM. También admite extraer esquemas de bases de datos existentes, lo que permite enfoques tanto database-first como code-first. Drizzle fue diseñado con los entornos serverless en mente: tiene un tamaño de paquete pequeño y admite sentencias preparadas (prepared statements), lo que permite que las sentencias SQL se pre-compilen para que el controlador de base de datos ejecute directamente SQL binario en lugar de analizar (parse) las consultas cada vez. Su simplicidad y compatibilidad con entornos serverless hacen que Drizzle sea una opción atractiva para el uso de ORMs dentro del ecosistema TypeScript.

105. Criptografía poscuántica en Java

Evaluar

Las computadoras cuánticas continúan avanzando rápidamente, con ofertas SaaS como AWS Braket que ahora proporcionan acceso a algoritmos cuánticos en múltiples arquitecturas.

Desde marzo, Java 24 ha introducido criptografía poscuántica en Java, agregando soporte para algoritmos criptográficos poscuánticos como ML-KEM y ML-DSA, y .Net 10 también ha ampliado su compatibilidad. Nuestro consejo es sencillo: si estás desarrollando software en estos lenguajes, comienza ahora a adoptar algoritmos seguros para la era cuántica, a fin de preparar tus sistemas para el futuro.

106. kagent

Evaluar

Kagent es un framework de código abierto para ejecutar inteligencia artificial basada en agentes dentro de clústeres de Kubernetes. Permite que los agentes basados en LLM planifiquen y ejecuten tareas operativas como diagnosticar problemas, corregir configuraciones o interactuar con herramientas de observabilidad mediante APIs nativas de Kubernetes e integraciones con Model Context Protocol (MCP). Su objetivo es llevar el concepto de "AgentOps" a la infraestructura cloud-native, combinando la gestión declarativa con el razonamiento autónomo.

Como proyecto del CNCF Sandbox, Kagent debe implementarse con precaución, especialmente por los riesgos asociados a otorgar capacidades de gestión operativa a los LLMs. Técnicas como el análisis de flujo tóxico cpueden ser especialmente útiles para evaluar y mitigar estos riesgos.

107. LangExtract

Evaluar

LangExtract es una biblioteca de Python que utiliza LLMs para extraer información estructurada de texto no estructurado según instrucciones definidas por el usuario. Procesa materiales específicos de dominio, como notas y reportes clínicos, identificando y organizando detalles clave mientras mantiene cada dato extraído trazable a su fuente. Las entidades extraídas pueden exportarse como un archivo `.jsonl` un formato estándar para datos de modelos de lenguaje, y visualizarse mediante una interfaz HTML interactiva para revisión contextual. Nuestros equipos evaluaron LangExtract para extraer entidades destinadas a poblar un grafo de conocimiento de dominio y lo encontraron eficaz para transformar documentos complejos en representaciones estructuradas y legibles por máquina.

108. Langflow

Evaluar

Langflow es una plataforma abierta de bajo código (low-code) para crear y visualizar flujos de trabajo con LLMs. Construida sobre LangChain, permite a los desarrolladores encadenar prompts, herramientas, bases de datos vectoriales y componentes de memoria mediante una interfaz de arrastrar y soltar, manteniendo al mismo tiempo compatibilidad con código Python personalizado para lógica avanzada. Es especialmente útil para crear prototipos de aplicaciones agénticas en agentes sin necesidad de escribir código completo de back-end. Sin embargo, Langflow aún es relativamente nuevo y presenta ciertas limitaciones para uso en producción. Nuestra advertencia habitual respecto a las plataformas de bajo código también aplica aquí. Dicho esto, nos gusta que los flujos de trabajo de Langflow puedan definirse y versionar como código, lo que ayuda a mitigar algunas de las desventajas de este tipo de plataformas.

109. LMCache

Evaluar

LMCache es una solución de caché de tipo clave-valor (KV) que acelera la infraestructura de despliegue e inferencia de LLMs. Actúa como una capa de almacenamiento en caché especializada sobre un conjunto de motores de inferencia con LLMs, almacenando entradas precomputadas para textos que probablemente se procesan varias veces, como historiales de chat o colecciones de documentos. Al persistir estos valores en disco, las operaciones de prefill pueden descargarse de la GPU, reduciendo el time-to-first-token (TTFT) y disminuyendo los costos de inferencia en cargas de trabajo exigentes como pipelines RAG, aplicaciones de chat de múltiples turnos y sistemas basados en agentes. Puedes integrar LMCache con servidores de inferencia principales como vLLM o NVIDIA Dynamo y creemos que vale la pena evaluar su impacto en tu configuración.

110. Mem0

Evaluar

Mem0 es una capa de memoria diseñada para agentes de inteligencia artificial. Los enfoques ingenuos suelen almacenar todo el historial de chat en una base de datos y reutilizarlo en conversaciones futuras, lo que genera un uso excesivo de tokens. Mem0 reemplaza este enfoque con una arquitectura más sofisticada que separa la memoria en una capa de recuerdo a corto plazo y una capa inteligente de largo plazo que extrae y almacena solo los hechos y relaciones más relevantes. Su arquitectura combina un almacén vectorial para la similitud semántica con un grafo de conocimiento para comprender datos temporales y relacionales. Este diseño reduce significativamente el uso de tokens de contexto, al tiempo que permite a los agentes mantener conciencia a largo plazo, algo sumamente útil para la personalización y muchos otros casos de uso.

111. Open Security Control Assessment Language (OSCAL)

Evaluar

Open Security Controls Assessment Language (OSCAL) es un formato de intercambio de información abierto y legible por máquina, diseñado para aumentar la automatización en la gestión del cumplimiento y el riesgo, y ayudar a los equipos a dejar atrás los enfoques manuales basados en texto. Liderado por el National Institute of Standards and Technology (NIST), OSCAL proporciona representaciones estándar en XML, JSON y YAML para expresar controles de seguridad asociados con marcos de la industria como SOC 2 y PCI, así como con marcos gubernamentales como FedRAMP en Estados Unidos, el Cybersecurity Control Catalogue de Singapur y el Information Security Manual de Australia.

Aunque OSCAL aún no ha sido ampliamente adoptado fuera del sector público y su ecosistema todavía está en proceso de maduración, nos entusiasma su potencial para optimizar las evaluaciones de seguridad, reducir la dependencia de hojas de cálculo y ejercicios de verificación, e incluso posibilitar el cumplimiento automatizado cuando se incorpora en plataformas de cumplimiento como código (compliance-as-code) y cumplimiento continuo.

112. OpenInference

Evaluar

OpenInference es un conjunto de convenciones y complementos; es complementario a OpenTelemetry y está diseñado para observar aplicaciones de IA. Proporciona instrumentación estandarizada para frameworks de aprendizaje automático y bibliotecas, lo que ayuda a las personas desarrolladoras a trazar las invocaciones de LLM junto con el contexto que las rodea, como recuperaciones desde almacenes vectoriales (vector stores) o llamadas a herramientas externas, API y motores de búsqueda. Los spans pueden exportarse a cualquier colector compatible con OTEL, garantizando la alineación con las pipelines de telemetría existentes. Anteriormente mencionamos Langfuse, una plataforma de observabilidad de LLM comúnmente utilizada; el SDK de OpenInference puede registrar trazas en Langfuse y en otras plataformas de observabilidad compatibles con OpenTelemetry.

113. Valibot

Evaluar

Valibot es una biblioteca de validación de esquemas en TypeScript. Al igual que otras bibliotecas populares de validación en TypeScript, como Zod y Ajv, proporciona inferencia de tipos, pero su diseño modular la diferencia. Esta arquitectura permite que los bundlers realicen tree shaking y code splitting de forma efectiva, incluyendo solo las funciones de validación que actualmente se utilizan. Valibot puede reducir el tamaño del paquete hasta en un 95% en comparación con Zod en escenarios óptimos. Es una opción atractiva para la validación de esquemas en entornos donde el tamaño del paquete es crítico, como la validación del lado del cliente o las funciones serverless.

114. Vercel AI SDK

Evaluar

Vercel AI SDK es un conjunto de herramientas de código abierto y de pila completa para crear aplicaciones y agentes impulsados por IA en el ecosistema de TypeScript. Consta de dos componentes principales: AI SDK Core, que estandariza las llamadas a LLM independientes del modelo y admite la generación de texto, la generación de objetos estructurados y el uso de herramientas; y AI SDK UI, que simplifica el desarrollo del front-end con transmisión, gestión de estado y actualizaciones de interfaz de usuario en tiempo real en React, Vue, Next.js y Svelte, de forma similar a assistant-ui. Para los equipos que ya trabajan dentro del ecosistema de TypeScript y Next.js, Vercel AI SDK ofrece una forma rápida y fluida de crear aplicaciones de IA con experiencias ricas del lado del cliente.

Mantente al día de todas las noticias y opiniones relacionadas con Radar

Suscríbete al Radar Tecnológico para recibir correos electrónicos cada dos meses con información sobre tecnología de Thoughtworks y futuras publicaciones del Radar Tecnológico.

Suscríbete ahora

Somos una consultoría tecnológica global que genera un impacto extraordinario al combinar experiencia en diseño, ingeniería e inteligencia artificial.

Durante más de 30 años, nuestra cultura de innovación y excelencia tecnológica ha ayudado a las organizaciones a fortalecer sus sistemas empresariales, escalar con agilidad y crear experiencias digitales fluidas.

Nos dedicamos a resolver los desafíos más críticos de nuestros clientes, combinando la inteligencia artificial y la creatividad humana para convertir ideas ambiciosas en realidad.

/thoughtworks

Diseño. Ingeniería. IA.