

# Radars Tecnológico

Una guía de opinión sobre el entorno tecnológico actual

Volumen 31  
Oct. 2024



/thoughtworks

Estrategia. Diseño. Ingeniería.

<b>Sobre el Radar</b>	<b>3</b>
<b>Un vistazo al Radar</b>	<b>4</b>
<b>Contribuyentes</b>	<b>5</b>
<b>Temas</b>	<b>7</b>
<b>El Radar</b>	<b>9</b>
<b>Técnicas</b>	<b>12</b>
<b>Plataformas</b>	<b>22</b>
<b>Herramientas</b>	<b>29</b>
<b>Lenguajes y Frameworks</b>	<b>41</b>

# 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

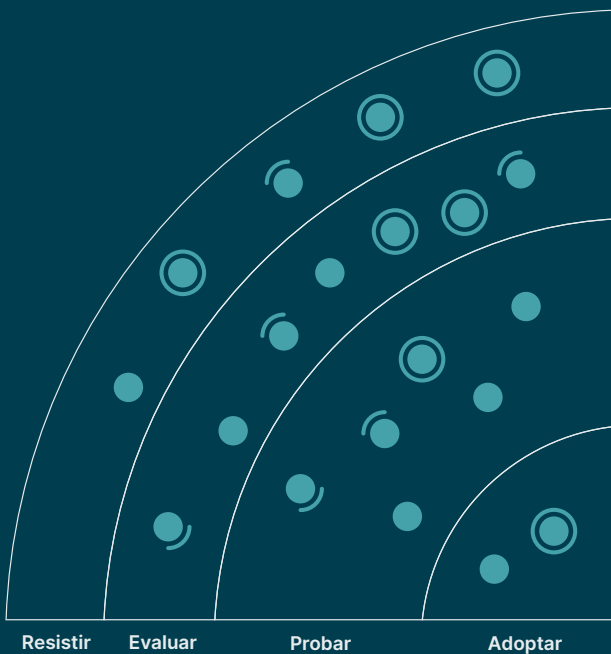
Para más información sobre el Radar, consulta [thoughtworks.com/es/radar/faq](https://www.thoughtworks.com/es/radar/faq).



# 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 adoptar 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 que 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 20 personas tecnológas 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 tecnológas en Thoughtworks. Esta edición del Radar Tecnológico de Thoughtworks es en base a la reunión virtual que TAB realizó remotamente en Septiembre del 2024.



Rachel Laycock  
(CTO)



Martin Fowler  
(Chief Scientist)



Rebecca Parsons  
(CTO Emerita)



Bharani Subramaniam



Birgitta Böckeler



Camilla Falconi Crispim



Erik Dörnenburg



James Lewis



Ken Mugrage



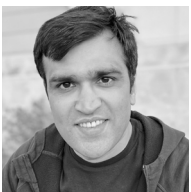
Maya Ormaza



Mike Mason



Neal Ford



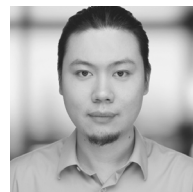
Pawan Shah



Scott Shaw



Selvakumar Natesan



Shangqi Liu



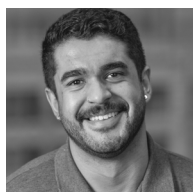
Sofia Tania



Thomas Squeo



Vanya Seth



Will Amaral

# Contribuyentes

## Equipo de traducción:

Mario Bizcocho, Elvira Segarra Ruiz, Carlos Barroso, Tex Albuja, Gabriela Marques, Eduard Maura i Puig, Jan Gomez Roberts, Sendami Luque Liñan, Alexandra Ortiz, Juan Romero, Andrés Renato Rojas, Ana López Estrella, Pablo Company Ramírez, Ana Corral, Antonia Castells, Antonio Galisteo, Ivanna Cevallos, Valeria Zaldumbide, Cristian Montero, Milber Champutiz, Gustavo Chiriboga, Andrea Astorga Bilbao, Luis Bustamante Flores, María Córdova, Miguel Vela Romera, Rocio Mera, Mafer Vilatuña, Jesús Cardenal, Núria Odena, Byron Torres, David Vellé Abel, Andrea Velez, Erika Vacacela, José Herdoíza, Micaela Díaz, Carlos Fuentes, Rodolfo Escobar, Catalina Solis, Elisa Torres, Jose María Alonso Montero, Jorge Agudo Praena, Najim Aghmir, Judit Navarro, Sebastian Roman, Maria Jaramillo, Cecilia Geraldo, Cesar Abreu, Nahuel Sotelo, Andrea Obando, Joseph Guerrero, Katherine Ayala, Iván González, Daniel Santibañez, Loreto Sánchez, Ximena Contreras y Gonzalo Martinez.

## Equipo de Edición:

Maya Ormaza, Fernando Tamayo y Nico Singh.

## Equipo de Marketing:

Elizabeth Parra.

## Antipatrones para generación de código con IA Generativa y LLM

No sorprende a nadie que la IA generativa y los LLM dominaran nuestras conversaciones para esta edición del Radar, incluyendo patrones emergentes en torno a su uso por parte de los desarrolladores. Los patrones inevitablemente conducen a antipatrones (situaciones contextualizadas que los desarrolladores deben evitar). Vemos algunos antipatrones que comienzan a aparecer en el espacio hiperactivo de la IA, incluida la creencia errónea de que los humanos pueden reemplazar completamente la programación en pares con la IA como compañera. Esto incluye la dependencia excesiva en las sugerencias de los asistentes de código, los problemas de calidad del código generado, y una tasa de crecimiento acelerada de las bases de código. La IA tiende a resolver problemas por fuerza bruta en lugar de utilizar abstracciones; tales como, usar docenas de condicionales apiladas en lugar del patrón de diseño de Estrategia. Los problemas de calidad del código resaltan especialmente la necesidad de una diligencia constante por parte de los desarrolladores y arquitectos para evitar sumergirse en código que funciona pero es deficiente. Por lo tanto, los miembros del equipo deben reforzar las buenas prácticas de ingeniería como pruebas unitarias, funciones de aptitud arquitectónica y otras técnicas probadas de gobernanza y validación para asegurarse de que la IA está ayudando a cumplir su objetivo, en lugar de complejizar su código.

## Rust: un lenguaje que sigue brillando

Rust se ha convertido gradualmente en el lenguaje de programación de sistemas preferido. En cada sesión del Radar, Rust surge una y otra vez en nuestras conversaciones; una gran cantidad de las herramientas que discutimos están escritas en Rust. Es el lenguaje de elección cuando se trata de reemplazar utilidades a nivel de sistema más antiguas, pero también para reescribir parte de un ecosistema con el fin de mejorar el rendimiento: el epíteto más común para las herramientas basadas en Rust parece ser “increíblemente rápido”. Por ejemplo, vemos varias herramientas en el ecosistema de Python que tienen alternativas basadas en Rust para ofrecer un rendimiento notablemente mejor. Los diseñadores del lenguaje y la comunidad han logrado crear un ecosistema muy apreciado de SDKs esenciales, bibliotecas y herramientas de desarrollo, proporcionando una velocidad de ejecución excepcional con menos inconvenientes que muchos de sus predecesores. Muchos en nuestro equipo son fanáticos de Rust, y parece que la mayoría de los desarrolladores que lo usan lo consideran muy valioso.

## **El crecimiento gradual de WASM**

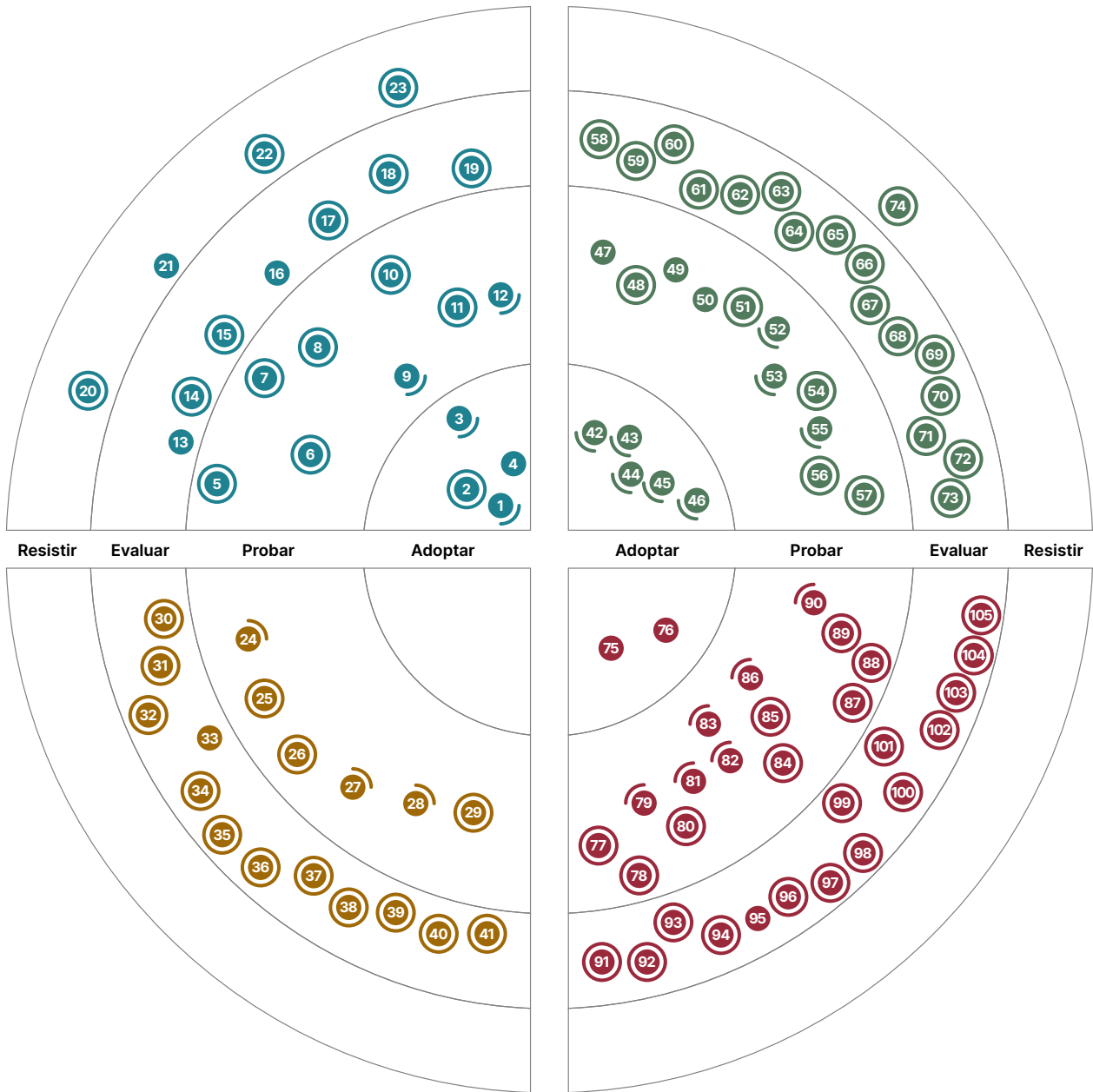
WASM (WebAssembly) es un formato de instrucciones binario para máquina virtual basada en pila, lo que suena esotérico y demasiado bajo nivel para el interés de la mayoría de las desarrolladoras hasta que la gente ve las implicaciones: la capacidad de ejecutar aplicaciones complejas dentro del sandbox del navegador. WASM puede ejecutarse dentro de las máquinas virtuales JavaScript existentes, permitiendo aplicaciones que las desarrolladoras sólo podían implementar anteriormente en frameworks nativos y extensiones que fueran empotrables dentro de los navegadores. Los cuatro navegadores principales soportan ahora WASM 1.0 (Chrome, Firefox, Safari y Edge), abriendo posibilidades emocionantes para un desarrollo sofisticado portable y multiplataforma. Hemos observado este estándar a lo largo de los últimos años con mucho interés y estamos contentos de ver cómo empieza a ejercitar sus capacidades como un objetivo de despliegue legítimo.




## **La explosión cámbrica de herramientas de IA generativa**

Siguiendo la trayectoria establecida en los últimos volúmenes del Radar, esperábamos que la IA generativa tuviera un lugar destacado en nuestras discusiones. Y, sin embargo, aún nos sorprendió la explosión en el ecosistema de tecnologías que soportan los modelos de lenguaje: guardrails, evals, herramientas para construir agentes, frameworks para trabajar con resultados estructurados, bases de datos vectoriales, servicios en la nube y herramientas de observabilidad. De muchas maneras, este crecimiento rápido y variado tiene perfecto sentido: la experiencia inicial, la simplicidad de una prompt en texto plano a un modelo de lenguaje, ha dado paso a la ingeniería de productos de software. Estos productos quizás no cumplan con los sueños y afirmaciones desmedidas que se hicieron después de que la gente enviara sus primeros prompts a ChatGPT, pero vemos un uso sensato y productivo de la IA generativa en muchos de nuestros clientes, y todas estas herramientas, plataformas y frameworks juegan un papel en llevar soluciones basadas en LLM a producción. Como sucedió con la explosión del ecosistema de JavaScript alrededor de 2015, esperamos que este crecimiento caótico continúe por un tiempo.



# El Radar



-  Nuevo
-  Desplazado adentro/afuera
-  Ningún cambio

# El Radar

## Técnicas

### Adoptar

1. 1% canary release
2. Pruebas de componentes
3. Despliegue continuo
4. Generación mejorada por recuperación (RAG)

### Probar

5. Narración de dominio
6. Fine-tuning a embeddings
7. Llamada a funciones con LLMs
8. LLM como juez
9. Llaves de acceso
10. Modelos de lenguaje pequeños
11. Datos sintéticos para pruebas y entrenamiento de modelos
12. Usar GenAI para entender las bases de código heredado

### Evaluar

13. Asistentes de IA para equipos
14. Prompting dinámico con pocas muestras
15. GraphQL para productos de datos
16. Agentes autónomos impulsados por modelos de lenguaje a gran escala (LLM)
17. Observabilidad 2.0
18. Inferencia con LLMs en dispositivos de usuario final
19. Salida estructurada de LLMs

### Resistir

20. Complacencia con el código generado por IA
21. Entornos de pruebas de integración para toda la empresa
22. Prohibiciones al uso de LLMs
23. Reemplazar codificación en parejas con IA

## Plataformas

### Adoptar

—

### Probar

24. Databricks Unity Catalog
25. FastChat
26. Constructor de Agentes de GCP Vertex AI
27. Langfuse
28. Qdrant
29. Vespa

### Evaluar

30. Azure AI Search
31. Databricks Delta Live Tables
32. Elasticsys Compliant Kubernetes
33. FoundationDB
34. Golem
35. Iggy
36. Iroh
37. Plataformas de modelos de visión de gran tamaño (LVM)
38. OpenBCI Galea
39. PGLite
40. SpinKube
41. Unblocked

### Resistir

—

## Adoptar

- 42. Bruno
- 43. K9s
- 44. SOPS
- 45. Herramientas para pruebas de regresión visual
- 46. Wiz

## Probar

- 47. AWS Control Tower
- 48. CCMenu
- 49. ClickHouse
- 50. Devbox
- 51. Diffastic
- 52. LinearB
- 53. pgvector
- 54. Herramienta de desarrollo de Snapcraft
- 55. Spinnaker
- 56. TypeScript OpenAPI
- 57. Unleash

## Evaluar

- 58. Astronomer Cosmos
- 59. ColPali
- 60. Cursor
- 61. Data Mesh Manager
- 62. GitButler
- 63. Asistente de IA de JetBrains
- 64. Mise
- 65. Mockoon
- 66. Raycast
- 67. ReadySet
- 68. Rspack
- 69. Semantic Router
- 70. Asistente de Ingeniería de Software
- 71. uv
- 72. Warp
- 73. Zed

## Resistir

- 74. CocoaPods

## Adoptar

- 75. dbt
- 76. Testcontainers

## Probar

- 77. CAP
- 78. CARLA
- 79. Databricks Asset Bundles
- 80. Instructor
- 81. Kedro
- 82. LiteLLM
- 83. LlamaIndex
- 84. LLM Guardrails
- 85. Medusa
- 86. PKI
- 87. ROS 2
- 88. seL4
- 89. SetFit
- 90. vLLM

## Evaluar

- 91. Apache XTable™
- 92. dbldatagen
- 93. DeepEval
- 94. DSPy
- 95. Flutter para la Web
- 96. kotaemon
- 97. Lenis
- 98. LLMingua
- 99. Microsoft Autogen
- 100. Pingora
- 101. Ragas
- 102. Score
- 103. shadcn
- 104. Slint
- 105. SST

## Resistir

—

# Técnicas

## Adoptar

1. 1% canary release
2. Pruebas de componentes
3. Despliegue continuo
4. Generación mejorada por recuperación (RAG)

## Probar

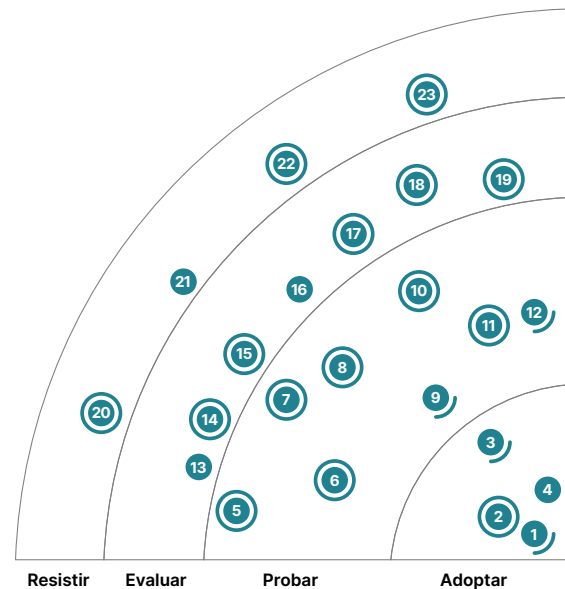
5. Narración de dominio
6. Fine-tuning a embeddings
7. Llamada a funciones con LLMs
8. LLM como juez
9. Llaves de acceso
10. Modelos de lenguaje pequeños
11. Datos sintéticos para pruebas y entrenamiento de modelos
12. Usar GenAI para entender las bases de código heredado

## Evaluar

13. Asistentes de IA para equipos
14. Prompting dinámico con pocas muestras
15. GraphQL para productos de datos
16. Agentes autónomos impulsados por modelos de lenguaje a gran escala (LLM)
17. Observabilidad 2.0
18. Inferencia con LLMs en dispositivos de usuario final
19. Salida estructurada de LLMs

## Resistir

20. Complacencia con el código generado por IA
21. Entornos de pruebas de integración para toda la empresa
22. Prohibiciones al uso de LLMs
23. Reemplazar codificación en parejas con IA



- Nuevo
- Desplazado dentro/afuera
- Ningún cambio

## 1. 1% canary release

### Adoptar

Durante muchos años, hemos utilizado el enfoque de canary release para fomentar el feedback temprano sobre nuevas versiones del software, mientras reducimos el riesgo mediante un despliegue incremental a usuarios seleccionados. El 1% canary es una técnica útil en la que desplegamos nuevas funcionalidades a un segmento muy pequeño (digamos, el 1%) de usuarios cuidadosamente seleccionados de varias categorías de usuarios. Esto permite a los equipos capturar rápidamente el feedback de los usuarios y observar el impacto de las nuevas versiones en aspectos como el rendimiento y la estabilidad, para aprender y responder según sea necesario. Esta técnica se vuelve especialmente crucial cuando los equipos están implementando actualizaciones de software en aplicaciones móviles o en una flota de dispositivos, como dispositivos de edge computing o vehículos definidos por software. Con una adecuada observabilidad y feedback temprano, ofrece la oportunidad de limitar el alcance del impacto en caso de que surjan escenarios inesperados en producción. Aunque las canary releases pueden ser útiles para obtener un feedback más rápido de los usuarios, creemos que comenzar con un pequeño porcentaje de los usuarios es obligatorio para reducir y contener el riesgo en despliegues de funcionalidades a gran escala.

## 2. Pruebas de componentes

### Adoptar

Las pruebas automatizadas siguen siendo la piedra angular del desarrollo eficaz de software. Para las pruebas de front-end podemos discutir si la distribución de distintos tipos de pruebas debería ser la clásica pirámide de pruebas o si debe tener forma de un trofeo. En cualquier caso, los equipos deberían centrarse en las pruebas de componentes porque los conjuntos de pruebas deben ser estables y ejecutarse rápidamente. En cambio, lo que estamos viendo es que los equipos renuncian a dominar las pruebas de componentes en favor de pruebas end-to-end basadas en el navegador, así como pruebas unitarias muy limitadas. Las pruebas unitarias tienden a forzar a los componentes a exponer lo que debería ser una funcionalidad puramente interna, mientras que las pruebas basadas en navegador son lentas, más inestables y más difíciles de depurar. Nuestra recomendación es tener una cantidad significativa de pruebas de componentes y utilizar una biblioteca como jsdom para ejecutar pruebas de componentes en memoria. Las herramientas de navegador como Playwright por supuesto, siguen teniendo un lugar en las pruebas de end-to-end, pero no deberían utilizarse para pruebas de componentes.

## 3. Despliegue continuo

### Adoptar

Creemos que las compañías deben adoptar prácticas de despliegue continuo siempre que sea posible. El despliegue continuo es la práctica de desplegar automáticamente a producción cada cambio que pasa las pruebas automatizadas. Esta práctica es un factor clave para lograr ciclos rápidos de retroalimentación y permite a las organizaciones entregar valor a los clientes de manera más rápida y eficiente. El despliegue continuo difiere de la entrega continua en que esta última sólo requiere que el código pueda ser desplegado en cualquier momento; no requiere que cada cambio realmente esté desplegado en producción. En el pasado habíamos dudado si mover el despliegue continuo hacia el anillo de Adopción, ya que es una práctica que requiere un alto nivel de madurez en otras áreas de la entrega de software y por lo tanto no es apropiada para todos los equipos. Sin embargo, el reciente libro de la Thoughtworker Valentina Servile Continuous Deployment proporciona una guía exhaustiva para implementar esta práctica en una organización. Ofrece un roadmap para que las organizaciones la sigan y así logren alcanzar el nivel de madurez requerido para adoptar prácticas de despliegue continuo.

## 4. Generación mejorada por recuperación (RAG)

### Adoptar

Generación mejorada por recuperación (RAG) es el patrón preferido por nuestros equipos para mejorar la calidad de las respuestas generadas por un modelo de lenguaje de gran tamaño (LLM). Lo hemos utilizado con éxito en muchos proyectos, incluyendo la [plataforma de IA Jugalbandi](#). Con RAG, la información sobre documentos relevantes y confiables se almacena en una base de datos. Para un prompt dado, se consulta la base de datos, se recuperan los documentos relevantes y se amplía el prompt con el contenido de los documentos, proporcionando así un contexto más completo al LLM. Esto resulta en una salida de mayor calidad y reduce considerablemente las alucinaciones. La ventana de contexto —que determina el tamaño máximo de la entrada del LLM— ha crecido significativamente con los modelos más nuevos, pero seleccionar los documentos más relevantes sigue siendo un paso crucial. Nuestra experiencia indica que un contexto más pequeño y cuidadosamente construido puede generar mejores resultados que un contexto amplio y grande. Utilizar un contexto grande también es más lento y costoso. Antes dependíamos únicamente de incrustaciones almacenadas en una base de datos vectorial para identificar el contexto adicional. Ahora estamos viendo re-ranking y búsqueda híbrida: herramientas de búsqueda como [Elasticsearch Relevance Engine](#) así también enfoques como [GraphRAG](#) que utilizan grafos de conocimiento creados con la ayuda de un LLM. El enfoque basado en grafos ha funcionado particularmente bien en nuestro trabajo sobre [la comprensión de bases de código heredado con GenAI](#).

## 5. Narración de dominio

### Probar

El diseño guiado por dominios (DDD) se ha convertido en un enfoque fundamental en la forma en que desarrollamos software. Lo utilizamos para modelar eventos, guiar diseños de software, establecer límites de contexto alrededor de microservicios y elaborar requisitos de negocio matizados. DDD establece un lenguaje ubicuo que tanto las partes interesadas no técnicas y los desarrolladores de software puedan usar para comunicarse efectivamente acerca del negocio. Una vez establecido, los modelos de dominio evolucionan, pero a muchos equipos les resulta difícil comenzar con DDD. No existe un enfoque único para construir un modelo de dominio inicial. Una técnica prometedora que hemos encontrado recientemente es la [narración de dominio \(domain storytelling\)](#). La narración de dominio es una técnica donde a los expertos en negocio se les solicita que describan las actividades en el negocio. A medida que los expertos son guiados a través de la narración, un facilitador usa un lenguaje pictográfico para capturar las relaciones y acciones entre las entidades y actores. El proceso de realizar estas historias ayuda visiblemente a clarificar y desarrollar un entendimiento compartido entre los participantes. Ya que no existe un único método correcto para desarrollar un modelo de dominio, la narración de dominio ofrece una alternativa digna de mención o, para un enfoque más completo de DDD, complementar con [Event Storming](#), que es otra técnica que utilizamos a menudo para empezar con DDD.

## 6. Fine-tuning a embeddings

### Probar

Al desarrollar aplicaciones LLM basadas en generación mejorada por recuperación (RAG por sus siglas en inglés), la calidad de los embeddings impacta directamente tanto en la recuperación de documentos relevantes como en la calidad de las respuestas. Aplicar fine-tuning a embeddings puede mejorar la precisión y relevancia de los embeddings para tareas o dominios específicos. Nuestros equipos hicieron fine-tuning a los embeddings al desarrollar aplicaciones LLM de dominios específicos, donde la extracción de información precisa es crucial. Sin embargo, hay que considerar las ventajas y desventajas de este enfoque antes de apresurarse a afinarlos.

## 7. Llamada a funciones con LLMs

### Probar

Llamada a funciones con LLMs se refiere a la capacidad de integrar los LLMs con funciones, APIs o herramientas externas; esto se logra identificando e invocando la función adecuada basada en una consulta dada y en la documentación asociada. Esta capacidad extiende la utilidad de los LLMs más allá de la generación de texto, permitiéndoles realizar tareas específicas como la recuperación de información, la ejecución de código y la interacción con APIs. Al invocar funciones externas o APIs, los LLMs pueden realizar acciones que antes estaban fuera de sus capacidades individuales. Esta técnica permite a los LLMs actuar sobre sus propios resultados, cerrando así la brecha entre el pensamiento y la acción, de manera similar a cómo los humanos usan herramientas para completar diversas tareas. Con la introducción de la llamada a funciones, los LLMs añaden determinismo y veracidad al proceso de generación, logrando un equilibrio entre creatividad y lógica. Este método permite a los LLMs conectarse a sistemas internos y bases de datos o incluso realizar búsquedas en internet a través de navegadores conectados. Modelos como la serie GPT de OpenAI admiten la llamada a funciones, y modelos perfeccionados como Gorilla están específicamente diseñados para mejorar la precisión y consistencia en la generación de llamadas a APIs ejecutables a partir de instrucciones en lenguaje natural. Como técnica, la llamada a funciones se sitúa dentro de la generación mejorada por recuperación (RAG) y arquitecturas de agentes. Debería verse como un patrón abstracto de uso, destacando su potencial como una herramienta fundamental en diversas implementaciones, más que como una solución específica.

## 8. LLM como juez

### Probar

Varios sistemas que construimos comparten dos importantes características: son capaces de responder una pregunta acerca de un conjunto de datos extenso y son casi imposibles de saber cómo se ha llegado a la solución. A pesar de esta opacidad nosotros aun queremos evaluar y mejorar la calidad de las respuestas. Con el patrón LLM como juez , nosotros usamos LLM para evaluar la respuesta de otro sistema, que a su vez podría estar basado en un LLM. Hemos visto este patrón ser usado para determinar la relevancia de los resultados de búsqueda en un catálogo de productos y evaluar si un chatbot basado en LLM estaba guiando a los usuarios en una dirección sensata. Naturalmente, el sistema evaluador debe estar configurado y calibrado de manera cuidadosa. Puede generar ganancias significativas en eficiencia, lo que, a su vez, se traduce en costos más bajos. Esta es una área de investigación en curso, con un estado actualizado y resumido en este artículo.

## 9. Passkeys

### Probar

Impulsadas por la alianza FIDO y respaldadas por Apple, Google y Microsoft, las passkeys están cerca de alcanzar un nivel de usabilidad masivo. Al configurar un nuevo inicio de sesión con llaves de acceso, se generan un par de claves: el sitio web recibe la clave pública y el usuario mantiene la clave privada. El proceso del inicio de sesión utiliza criptografía asimétrica. El usuario demuestra que está en posesión de la clave privada, que se almacena en el dispositivo del usuario y nunca se envía al sitio web. El acceso a las llaves está protegido mediante biometría o un PIN. Las llaves de acceso pueden almacenarse y sincronizarse en los ecosistemas de las grandes tecnológicas, utilizando iCloud Keychain de Apple, Google Password Manager o Windows Hello. Para usuarios multiplataforma, el Client to Authenticator Protocol (CTAP) permite que las llaves de acceso se guarden en un dispositivo diferente al que crea la clave o la necesita para el inicio de sesión. La objeción más común al uso de las passkeys es que representan un desafío para los usuarios con menos conocimientos técnicos, lo cual creemos que es una visión contraproducente. Estos suelen ser los mismos usuarios que tienen una gestión deficiente de las contraseñas y, por lo tanto, serían los más beneficiados por métodos alternativos. En la práctica, los sistemas que usan llaves de acceso pueden recurrir a métodos de autenticación tradicionales si es necesario.

## 10. Modelos de lenguaje pequeños

### Probar

Los modelos de lenguaje de gran tamaño (LLM) han demostrado su utilidad en muchas áreas de aplicación, pero el hecho de que sean grandes puede ser una fuente de problemas: responder a una consulta requiere muchos recursos de cómputo, lo que hace que las consultas sean lentas y caras; los modelos son propietarios y tan grandes que deben ser alojados en una nube por un tercero, lo que puede ser problemático para los datos sensibles; y entrenar un modelo es excesivamente caro en la mayoría de los casos. El último problema puede resolverse con el patrón RAG, que evita la necesidad de entrenar y afinar los modelos básicos, pero los problemas de costo y privacidad suelen persistir. Por ello, cada vez hay más interés en los modelos de lenguaje pequeños (SLM). En comparación con sus hermanos más populares, tienen menos pesos y menos precisión, normalmente entre 3,5 y 10B parámetros. Investigaciones recientes sugieren que, en el contexto adecuado y si se configuran correctamente, los SLM pueden rendir o incluso superar a los LLM. Y su tamaño permite ejecutarlos en dispositivos periféricos. Ya hemos mencionado el Gemini Nano, de Google, pero el panorama está evolucionando rápidamente, con Microsoft presentando su serie Phi-3, por ejemplo.

## 11. Datos sintéticos para pruebas y entrenamiento de modelos

### Probar

La creación de sets de datos sintéticos implica generar datos artificiales que puedan imitar escenarios del mundo real sin depender de fuentes de datos sensibles o de acceso limitado. Aunque los datos sintéticos para sets de datos estructurados se han explorado ampliamente (por ejemplo, para pruebas de rendimiento o entornos seguros para la privacidad), estamos viendo un uso renovado de los datos sintéticos para datos no estructurados. A menudo, las empresas se enfrentan a la falta de datos etiquetados específicos del dominio, especialmente para su uso en el entrenamiento o el ajuste de los LLM. Herramientas como Bonito y Microsoft's AgentInstruct pueden generar datos sintéticos de ajuste de instrucciones a partir de fuentes crudas como documentos de texto y archivos de código. Esto ayuda a acelerar el entrenamiento del modelo al tiempo que reduce los costes y la dependencia de la curación manual de datos. Otro caso de uso importante es la generación de datos sintéticos



para tratar datos desequilibrados o dispersos, algo habitual en tareas como la detección de fraudes o la segmentación de clientes. Técnicas como SMOTE ayudan a equilibrar conjuntos de datos creando artificialmente instancias de clases minoritarias. Del mismo modo, en sectores como el financiero, las redes generativas adversariales (GAN) se utilizan para simular transacciones poco frecuentes, lo que permite que los modelos sean robustos a la hora de detectar casos extremos y mejorar el rendimiento general.

## 12. Usar GenAI para entender las bases de código heredado

### Probar

La IA Generativa (GenAI) y los modelos de lenguaje de gran tamaño (LLMs) pueden ayudar a los desarrolladores a escribir código y entenderlo. Esta ayuda es especialmente útil en el caso de bases de código heredado con documentación deficiente, incompleta y/o desactualizada. Desde nuestra última actualización sobre este tema, las técnicas y herramientas sobre el usar GenAI para entender las bases de código heredado han evolucionado significativamente. Hemos utilizado con éxito algunas de estas técnicas en la práctica, especialmente para apoyar esfuerzos de ingeniería inversa en la modernización de mainframes. Una técnica especialmente prometedora que hemos utilizado es un enfoque de generación aumentada por recuperación (RAG) en el cual la recuperación de información se realiza a partir de un grafo de conocimiento del código. Este grafo puede conservar información estructural sobre la base de código que va más allá de lo que un modelo de lenguaje de gran tamaño (LLM) podría extraer únicamente del código textual. Esto resulta particularmente útil en bases de código heredado que son menos autodescriptivas y cohesivas. Además, hay una oportunidad adicional para mejorar la comprensión del código, ya que el grafo puede enriquecerse aún más con documentación existente y generada por IA, dependencias externas, conocimientos del dominio de negocio o cualquier otro recurso disponible que facilite el trabajo de la IA.

## 13. Asistentes de IA para equipos

### Evaluar

Las herramientas de asistencia de codificación de IA se mencionan principalmente en el contexto de la asistencia y la mejora del trabajo de un colaborador individual. Sin embargo, la entrega de software es y seguirá siendo un trabajo en equipo, por lo que se debería buscar formas de crear asistentes de IA para equipos que ayuden a crear el equipo 10x, en lugar de un grupo aislado de ingenieros 10x asistidos por IA. Afortunadamente, los recientes desarrollos en el mercado de herramientas nos acercan a hacer de esto una realidad. Unblocked es una plataforma que reúne todas las fuentes de conocimiento de un equipo y las integra de forma inteligente en las herramientas de los miembros del equipo. Rovo de Atlassian incorpora la IA a la plataforma más utilizada de colaboración en equipo, brindando a los equipos nuevos tipos de búsqueda y acceso a su documentación, además de desbloquear nuevas formas de automatización y soporte a prácticas de software con agentes Rovo. Mientras esperamos que el mercado siga evolucionando en este ámbito, hemos estado explorando el potencial de la IA para la amplificación del conocimiento y el soporte a las prácticas de equipo: abrimos nuestro asistente de equipo Haiven y comenzó a recopilar aprendizajes con asistencia de IA para tareas no asociadas a la programación, como el análisis de requerimientos.

## 14. Prompting dinámico con pocas muestras

### *Evaluar*

El prompting dinámico con pocas muestras se basa en el prompting con pocas muestras incluyendo de forma dinámica ejemplos específicos en la instrucción (prompt) para guiar las respuestas del modelo. Ajustar el número y relevancia de estos ejemplos optimiza la extensión del contexto y su relevancia, mejorando así la eficiencia y el rendimiento del modelo. Hay librerías de código, como scikit-llm que implementan esta técnica usando la búsqueda del vecino más cercano para encontrar los ejemplos más relevantes de acuerdo con la consulta del usuario. Esta técnica permite hacer un mejor uso del limitado marco contextual del modelo y reducir el consumo de tokens. El generador de código SQL de código libre vanna utiliza prompting dinámico con pocas muestras para mejorar la precisión de sus resultados.

## 15. GraphQL para productos de datos

### *Evaluar*

GraphQL para productos de datos es la técnica de usar GraphQL como un punto de acceso para los productos de datos para que clientes consuman el producto. Hemos hablado acerca de GraphQL como un protocolo de API y cómo permite a desarrolladores crear una capa unificada de API que abstrae la complejidad de los datos subyacentes, entregando una interfaz más cohesionada y manejable a los clientes. GraphQL para productos de datos facilita a los consumidores la tarea de descubrir el formato y relaciones de los datos con el esquema de GraphQL y usar herramientas de cliente familiares para GraphQL. Nuestros equipos están explorando esta técnica en casos de uso específicos como talk-to-data para explorar y descubrir insights de big data con la ayuda de grandes modelos de lenguaje (LLMs por sus siglas en inglés) donde las consultas de GraphQL están construidas por LLMs basadas en los prompts del usuario y el esquema de GraphQL es usado en los prompts de la LLM como referencia.

## 16. Agentes autónomos impulsados por modelos de lenguaje a gran escala (LLM)

### *Evaluar*

Los agentes autónomos impulsados por modelos de lenguaje a gran escala (LLM) están evolucionando más allá de agentes individuales y sistemas multiagente estáticos con la aparición de frameworks como Autogen y CrewAI. Esta técnica permite a los desarrolladores descomponer una actividad compleja en varias tareas más pequeñas, realizadas por agentes a los que se les asigna un rol específico. Los desarrolladores pueden utilizar herramientas preconfiguradas para ejecutar las tareas, mientras que los agentes se comunican entre sí y orquestan el flujo de trabajo. La técnica aún se encuentra en sus primeras etapas de desarrollo. En nuestros experimentos hasta ahora, nuestros equipos han encontrado problemas como agentes que entran en bucles continuos y comportamientos descontrolados. Librerías como LangGraph ofrecen un mayor control sobre las interacciones de los agentes, permitiendo definir el flujo como un gráfico. Si decides utilizar esta técnica, sugerimos implementar mecanismos a prueba de fallos, como límites de tiempo y supervisión humana.

## 17. Observabilidad 2.0

### *Evaluar*

Observabilidad 2.0 representa un cambio de las herramientas de monitoreo tradicionales y dispersas hacia un enfoque unificado que aprovecha datos estructurados y de alta cardinalidad de eventos en un único repositorio de datos. Este modelo captura eventos enriquecidos y sin procesar con metadatos detallados para proporcionar una fuente de verdad única para un análisis exhaustivo. Al almacenar los eventos en su forma cruda, simplifica la correlación y soporta el análisis forense y en tiempo real, permitiendo obtener una visión más profunda sobre sistemas distribuidos complejos. Este enfoque permite un monitoreo de alta resolución y capacidades de investigación dinámicas. Observabilidad 2.0 da prioridad a la captura de datos con alta cardinalidad y alta dimensión, permitiendo un análisis detallado sin cuellos de botella de rendimiento. El repositorio de datos unificado reduce la complejidad, ofreciendo una visión coherente del comportamiento del sistema y alineando las prácticas de observabilidad más estrechamente con el ciclo de vida del desarrollo de software.

## 18. Inferencia con LLMs en dispositivos de usuario final

### *Evaluar*

Los modelos de lenguaje de gran tamaño o LLMs (siglas en inglés para Large Language Model) ahora son capaces de correr en navegadores web y dispositivos de usuario final, como teléfonos inteligentes y computadores portátiles, permitiendo que aplicaciones de AI se ejecuten en el dispositivo. Esto permite el manejo seguro de datos sensibles sin necesidad de transferir datos hacia la nube, muy baja latencia en tareas como edge computing y procesamiento de imagen o video en tiempo real, costos reducidos al realizar cómputos localmente y mantener funcionalidad incluso cuando no se cuenta con una conexión estable a internet. Ésta es un área de continua investigación y desarrollo. En ediciones pasadas mencionamos [MLX](#), un framework de código abierto para machine learning eficiente en procesadores Apple silicon. Otras herramientas que están emergiendo incluyen [Transformers.js](#) y [Chatty](#). Transformers.js nos permite correr Transformers en el navegador usando el ONNX Runtime, soportando modelos convertidos desde como PyTorch, TensorFlow y JAX. Chatty se apalanca en WebGPU para correr LLMs de forma nativa y privada en el navegador, ofreciendo una experiencia de AI enriquecida dentro del mismo.

## 19. Salida estructurada de LLMs

### *Evaluar*

La salida estructurada de LLMs se refiere a la práctica de restringir la respuesta de un modelo de lenguaje, a un esquema definido. Esto se puede lograr ya sea a través de instruir a un modelo generalizado que responda en un formato particular o realizando fine-tuning a un modelo para obtener una salida "nativa", por ejemplo, JSON. OpenAI ahora soporta salida estructurada, permitiendo a los desarrolladores proporcionar un esquema JSON, [pydantic](#) o un objeto Zod para limitar las respuestas de un modelo. Esta capacidad es particularmente valiosa ya que permite llamadas a funciones, interacciones con una API e integraciones externas, donde la precisión y el cumplimiento de un formato son críticas. La salida estructurada no solo mejora la forma en que los LLMs pueden interactuar con el código, sino que también soporta un mayor cantidad de casos de uso, como generación de markup para el renderizado de gráficos. Adicionalmente, la salida estructurada ha demostrado reducir las posibilidades de alucinaciones en la salida de un modelo.

## 20. Complacencia con el código generado por IA

### Resistir

Los asistentes de código con IA como [GitHub Copilot](#) y [Tabnine](#) se han hecho muy populares. Según la [encuesta para desarrolladores de StackOverflow de 2024](#), “el 72% de todos los encuestados están a favor o muy a favor de las herramientas con IA para el desarrollo”. Aunque también vemos sus ventajas, desconfiamos del impacto a medio y largo plazo que esto tendrá en la calidad del código y advertimos a los desarrolladores sobre la complacencia con el código generado por IA. Tras unas cuantas experiencias positivas con un asistente, resulta demasiado tentador ser permisivos al revisar las sugerencias de la IA. Estudios como [éste](#), de [GitClear](#) muestran una tendencia a un crecimiento más rápido de las bases de código, que [sospechamos coincide](#) con un mayor número de pull requests. Y [este estudio de GitHub](#) nos hace preguntarnos si el mencionado aumento del 15% de la tasa de fusión de pull requests es realmente algo bueno o si la gente está fusionando pull requests más grandes más rápido porque confían demasiado en los resultados de la IA. Seguimos utilizando los consejos básicos [“para empezar” que dimos hace más de un año](#), que consisten en tener cuidado con el sesgo de automatización, la falacia del costo hundido, el sesgo de anclaje y la fatiga de revisión. También recomendamos que los programadores desarrollen un buen [framework mental sobre dónde y cuándo no utilizar o confiar en la IA.](#)

## 21. Entornos de pruebas de integración para toda la empresa

### Resistir

Crear entornos de pruebas de integración para toda la empresa es una práctica frecuente y costosa que lo ralentiza todo. Estos entornos invariablemente se convierten en recursos valiosos que son difíciles de replicar y un cuello de botella para el desarrollo. También proveen de una falsa sensación de seguridad por sus inevitables discrepancias en los datos y configuraciones entre entornos. Irónicamente, la objeción habitual a sus alternativas — ya sean entornos efímeros o múltiples entornos de prueba locales — es su costo. Sin embargo, no se considera el costo de los retrasos causados por los entornos de integración para toda la empresa, como tener equipos de desarrollo en espera de que otros equipos terminen, o a los despliegues de nuevas versiones de sistemas dependientes. En su lugar, los equipos deberían usar entornos efímeros y, preferiblemente, un conjunto de pruebas propias del equipo de desarrollo que puedan ser ejecutadas y descartadas a bajo costo, usando stubs para sus sistemas en vez de réplicas reales. Para otras técnicas que soportan esta alternativa echa un vistazo a [pruebas de contrato guiados por el consumidor](#), [desacoplar el despliegue de la publicación](#), [centrarse en el tiempo medio de recuperación y pruebas en producción](#).

## 22. Prohibiciones al uso de LLMs

### Resistir

En lugar de imponer prohibiciones al uso de LLMs en el lugar de trabajo, las organizaciones deberían enfocarse en proveer acceso a un conjunto de herramientas de IA aprobadas. Una prohibición sólo empuja a los empleados a encontrar soluciones alternativas no aprobadas y potencialmente inseguras, creando riesgos innecesarios. Como en los días primarios de la computadora personal, la gente va a utilizar cualquier herramienta que sienta efectiva para hacer su trabajo, a pesar de las barreras impuestas. Al no proveer una alternativa segura y respaldada, las compañías corren el riesgo

de que los empleados utilicen LLM no aprobados que vienen acompañados de riesgos de propiedad intelectual, fuga de datos y responsabilidad. Por otro lado, ofrecer LLMs o herramientas de IA seguras y aprobadas por la empresa, garantiza la seguridad y productividad. Un enfoque bien gobernado permite a las organizaciones manejar sus inquietudes sobre privacidad de datos, seguridad, cumplimiento y costos, al mismo tiempo que empodera a los empleados con las capacidades que los LLMs ofrecen. En el mejor de los casos, un acceso bien manejado a herramientas IA puede acelerar el aprendizaje organizacional acerca de las mejores maneras de usar IA para el trabajo.

## 23. Reemplazar codificación en parejas con IA

### Resistir

Cuando se habla de codificación en pares, inevitablemente surge el tema de pair programming. Nuestra profesión tiene una relación de amor-odio con ella: algunos la adoran, otros no la soportan. Los codificadores en pares plantean ahora la siguiente pregunta: ¿puede un humano trabajar en par con la IA, en lugar de con otro humano, y obtener los mismos resultados para el equipo? GitHub Copilot incluso se llama a sí mismo «tu codificador en parejas de IA». Aunque creemos que un asistente de programación puede aportar algunas de las ventajas de la programación en parejas, desaconsejamos totalmente reemplazar la programación en parejas por la IA. Enmarcar a los asistentes de programación como codificadores en pareja ignora uno de los principales beneficios de la programación en pareja: mejorar el equipo, no sólo a los colaboradores individuales. Los asistentes de programación pueden ser útiles para desbloquearse, aprender sobre una nueva tecnología, incorporarse o agilizar el trabajo táctico para que podamos centrarnos en el diseño estratégico. Pero no contribuyen a ninguno de los beneficios de la colaboración en equipo, como mantener bajo el trabajo en curso, reducir los traspasos y el re-aprendizaje, hacer posible la integración continua o mejorar la propiedad colectiva del código.

# Plataformas

## Adoptar

---

## Probar

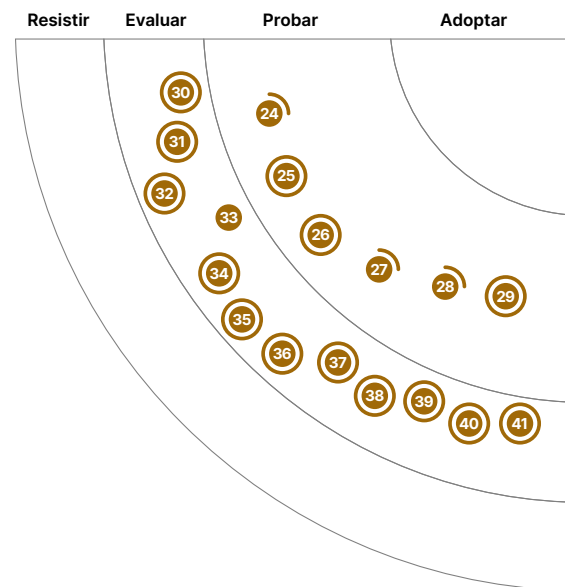
- 24. Databricks Unity Catalog
- 25. FastChat
- 26. Constructor de Agentes de GCP Vertex AI
- 27. Langfuse
- 28. Qdrant
- 29. Vespa

## Evaluar

- 30. Azure AI Search
- 31. Databricks Delta Live Tables
- 32. Elasticsys Compliant Kubernetes
- 33. FoundationDB
- 34. Golem
- 35. Iggy
- 36. Iroh
- 37. Plataformas de modelos de visión de gran tamaño (LVM)
- 38. OpenBCI Galea
- 39. PGLite
- 40. SpinKube
- 41. Unblocked

## Resistir

---



- Nuevo
- Desplazado dentro/afuera
- Ningún cambio

## 24. Databricks Unity Catalog

### Probar

Databricks Unity Catalog es una solución de gobernanza de datos para recursos, tales como archivos, tablas o modelos de machine learning presentes en un lakehouse. Es una versión administrada de la solución open-source Unity Catalog que puede ser usada para administrar y consultar datos guardados externamente o bajo el entorno de Databricks. Nuestros equipos han trabajado con una variedad de soluciones para administración de datos en el pasado, tales como Hive metastore o Microsoft Purview. Sin embargo, el soporte combinado de Unity Catalog para gobernanza, administración de metastore y data discovery la hacen atractiva, debido a que reduce la necesidad de administrar diferentes herramientas. Una complicación que nuestro equipo pudo descubrir en Unity Catalog administrada por Databricks, fue la falta de recuperación de desastre automática. Si bien se pudo configurar una funcionalidad propia de respaldo y restauración, habría sido más conveniente tener una solución provista por Databricks. Hay que tener en cuenta que a pesar de que estas plataformas de gobernanza usualmente implementan una solución centralizada para asegurar la consistencia entre workspaces y workloads, la responsabilidad de gobernar puede ocurrir de forma federada, habilitando a equipos individuales a administrar sus propios recursos.

## 25. FastChat

### Probar

FastChat es una plataforma abierta para entrenar, servir y evaluar grandes modelos de lenguaje. Nuestros equipos usan la funcionalidad de servir modelos para alojar distintos modelos — Llama 3.1 (8B and 70B), Mistral 7B y Llama-SQL — con distintos propósitos, todos en el formato consistente de OpenAI API. FastChat opera en una arquitectura ejecutor-controlador, permitiendo a múltiples ejecutores alojar modelos diferentes. Soporta tipos de ejecutores con vLLM, LiteLLM y MLX. Nosotros usamos los ejecutores del modelos vLLM por su alto rendimiento. Dependiendo del caso de uso — latencia o rendimiento — se pueden crear o escalar distintos tipos de ejecutores del modelo FastChat. Por ejemplo, el modelo usado para sugerencias de código en los entornos de desarrollo requiere una latencia baja y puede escalarse con múltiples ejecutores de FastChat para gestionar peticiones concurrentes eficientemente. Por el contrario, el modelo usado para Text-to-SQL no necesita muchos ejecutores por su baja demanda o requisitos de rendimiento distintos. Nuestros equipos se valen de la escalabilidad de FastChat para hacer pruebas A/B. Configuramos ejecutores FastChat con el mismo modelo pero con distintos valores para los hiper parámetros y planteamos las mismas preguntas a cada uno, identificando así hiper parámetros óptimos. Cuando migramos modelos de servicios en producción, llevamos a cabo pruebas A/B para asegurar que no haya interrupciones del servicio. Por ejemplo, migramos hace poco de CodeLlama 70B a Llama 3.1 70B para sugerencias de código. Ejecutando ambos modelos concurrentemente y comparando sus salidas, verificamos que el nuevo modelo cumplía o excedía el rendimiento del modelo anterior sin afectar la experiencia de desarrollo.

## 26. Constructor de Agentes de GCP Vertex AI

### Probar

El Constructor de Agentes de GCP Vertex AI proporciona una plataforma flexible para crear agentes de IA utilizando tanto lenguaje natural como un enfoque basado en código. Esta herramienta se integra perfectamente con datos de negocio a través de conectores de terceros y tiene todas las herramientas necesarias para construir, prototipar y desplegar agentes de IA. A medida que crece la necesidad de agentes de IA, muchos equipos tienen problemas para entender sus beneficios e implementación. Por eso, el Constructor de Agentes de GCP Vertex AI facilita a los desarrolladores

el prototipado rápido de agentes, así como el manejo de tareas de datos complejas con una mínima configuración. Nuestros desarrolladores lo han encontrado particularmente útil para construir sistemas basados en agentes, como bases de conocimiento o sistemas de soporte automatizados, que manejan eficientemente tanto datos estructurados como no estructurados, lo que lo convierte en una valiosa herramienta para desarrollar soluciones basadas en IA.

## 27. Langfuse

### *Probar*

Los LLMs funcionan como cajas negras, lo que dificulta determinar sus comportamientos. La observabilidad es crucial para abrir esta caja negra y comprender cómo las aplicaciones LLM operan en producción. Nuestros equipos han tenido experiencias positivas con Langfuse observando, monitoreando y evaluando aplicaciones basadas en LLMs. Su trazabilidad, análisis y capacidades de evaluación nos permiten analizar el rendimiento y precisión, administrar costos y latencia, y comprender patrones usados en producción, como por ejemplo, facilitando continuas mejoras basadas en datos. Los datos de instrumentación proporcionan una trazabilidad completa del flujo de petición-respuesta y de los pasos intermedios, la cual puede ser usada para datos de prueba, validando la aplicación antes de desplegar los cambios. Nosotros hemos utilizado Langfuse con RAG (generación mejorada por recuperación), entre otras arquitecturas de LLM, y agentes autónomos impulsados por LLMs. En una aplicación basada en RAG, por ejemplo, analizar trazas de conversaciones con puntuación baja, ayuda a identificar qué partes de la arquitectura - pre-entrenado, entrenado o generación - necesitan refinamiento. Otra opción que merece la pena considerar en este ámbito es Langsmith.

## 28. Qdrant

### *Probar*

Qdrant es un motor de búsqueda de similitud vectorial y base de datos de código abierto escrito en Rust. Es compatible con una amplia y densa gama de modelos de incrustación vectorial texto multimodales. Nuestros equipos han utilizado modelos de incrustación de código abierto como MiniLM-v6 y BGE para múltiples bases de conocimiento de productos. Utilizamos Qdrant como almacén vectorial empresarial con multi-tenancy para almacenar incrustaciones vectoriales como colecciones separadas, aislando la base de conocimiento de cada producto en el almacenamiento. Las políticas de acceso de los usuarios se gestionan en la capa de aplicación.

## 29. Vespa

### *Probar*

Vespa es un motor de búsqueda de código abierto y una plataforma de procesamiento de big data. Es especialmente adecuado para aplicaciones que requieren baja latencia y alto rendimiento. A nuestros equipos les gusta la capacidad de Vespa para implementar búsquedas híbridas con múltiples técnicas de recuperación, filtrar y ordenar eficazmente muchos tipos de metadatos, implementar ranking multifase, indexar múltiples vectores (por ejemplo, para cada chunk) por documento sin duplicar todos los metadatos en documentos indexados por separado y recuperar datos de múltiples campos indexados a la vez.



## 30. Azure AI Search

### *Evaluar*

Azure AI Search, anteriormente conocido como Cognitive Search, es un servicio de búsqueda basado en la nube diseñado para manejar datos estructurados y no estructurado como bases de conocimientos, particularmente en configuraciones de retrieval augmented generation(RAG) Soporta varios tipos de búsqueda, como búsqueda por palabras clave, vectorial e híbrida, lo que consideramos que será cada vez más importante. El servicio es capaz de ingerir automáticamente formatos de datos no estructurados de los más comunes, como PDF, DOC y PPT, lo que agiliza el proceso de creación de contenidos con capacidad de búsqueda. Además, se integra con otros servicios de Azure, como Azure OpenAI, permitiendo a los usuarios construir aplicaciones con un esfuerzo de integración manual mínimo. Desde nuestra experiencia, Azure AI Search funciona de forma fiable y es idónea para proyectos alojados en el entorno Azure. A través de sus capacidades personalizadas, los usuarios pueden adicionalmente definir pasos específicos para el procesamiento de datos. En general, si trabajas en el ecosistema Azure y necesitas una solución de búsqueda robusta para una aplicación RAG, vale la pena considerar Azure AI Search.

## 31. Databricks Delta Live Tables

### *Evaluar*

Databricks Delta Live Tables es un framework declarativo diseñado para construir pipelines de procesamiento de datos que sean confiables, mantenibles y comprobables. Permite a los ingenieros de datos definir transformaciones de datos utilizando un enfoque declarativo y gestiona automáticamente la infraestructura subyacente y el flujo de datos. Una de las características más destacadas de Delta Live Tables es su robusta capacidad de monitoreo. Proporciona un Grafo Acíclico Dirigido (DAG) de todo tu pipeline de datos, representando visualmente el movimiento de datos desde la fuente hasta las tablas finales. Esta visibilidad es crucial para pipelines complejas, ayudando a los ingenieros de datos y a los científicos de datos a rastrear la procedencia y las dependencias de los datos. Delta Live Tables está profundamente integrado en el ecosistema de Databricks, lo que también presenta algunos desafíos para personalizar las interfaces. Recomendamos a los equipos evaluar cuidadosamente la compatibilidad de las interfaces de entrada y salida antes de usar Delta Live Tables

## 32. Elastisys Compliant Kubernetes

### *Evaluar*

Elastisys Compliant Kubernetes es una distribución especializada de Kubernetes que está diseñada para cumplir con requerimientos regulatorios y de cumplimiento rigurosos, particularmente para organizaciones que operan en industrias altamente reguladas tales como salud, finanzas o gobierno. Tiene procesos de seguridad automatizados, proporciona soporte a entornos de múltiples nubes (multicloud) y on-premise y está construido sobre una arquitectura de seguridad de confianza cero. El énfasis en cumplimiento incorporado para leyes tales como GDPR y HIPAA y controles como ISO27001 la convierte en una opción atractiva para las empresas que necesiten un entorno de Kubernetes seguro, conforme a regulaciones y confiable.

## 33. FoundationDB

### *Evaluar*

FoundationDB es una base de datos multimodelo adquirida por Apple en 2015 y luego liberada como código abierto en abril de 2018. El núcleo de FoundationDB es un almacén distribuido de clave-valor, lo cual proporciona transacciones con serialización estricta. Desde que lo mencionamos por primera vez en el Radar, ha experimentado mejoras significativas —incluyendo distribuciones de datos inteligentes para evitar hotspots de escritura, un nuevo motor de almacenamiento, optimizaciones de rendimiento y soporte para replicación en múltiples regiones. Estamos utilizando FoundationDB en uno de nuestros proyectos en curso y estamos muy impresionados con su arquitectura modular. Esta arquitectura nos permite escalar diferentes partes del clúster de manera independiente. Por ejemplo, podemos ajustar el número de registros de transacciones, servidores de almacenamiento y proxies en función de nuestra carga de trabajo y hardware específicos. A pesar de sus extensas funcionalidades, FoundationDB sigue siendo notablemente sencillo de ejecutar y operar en grandes clústers.

## 34. Golem

### *Evaluar*

La computación duradera, un movimiento reciente en computación distribuida, usa un estilo de arquitectura de máquina de estados explícita para persistir la memoria de servidores serverless para una mejor tolerancia a fallas y recuperación. Golem es uno de los promotores de este movimiento. El concepto puede funcionar en algunos escenarios, como sagas de microservicios de larga duración o flujos de trabajo de larga duración en la orquestación de agentes de IA. Hemos evaluado Temporal en radares anteriores para fines similares y Golem es otra opción. Con Golem puedes escribir componentes WebAssembly en cualquier lenguaje compatible, además de que Golem es determinista y admite tiempos de inicio rápidos. Creemos que Golem es una plataforma emocionante que vale la pena evaluar.

## 35. Iggy

### *Assess*

Iggy, una plataforma de streaming de mensajes con persistencia escrita en Rust, es un proyecto relativamente nuevo con características interesantes. Soporta características como múltiples streams, tópicos y particiones, máximo una entrega, expiración de mensajes y soporte para TLS sobre los protocolos QUIC, TCP y HTTP. Iggy se ejecuta en un único servidor ofreciendo un alto rendimiento para operaciones de lectura y escritura. Con el próximo soporte a clusterización y a io\_uring, Iggy puede ser una alternativa potencial a Kafka.

## 36. Iroh

### *Evaluar*

Iroh es un sistema de almacenamiento de archivos distribuido y entrega de contenido relativamente nuevo, diseñado como una evolución de sistemas descentralizados existentes como IPFS (InterPlanetary File System). Tanto Iroh como IPFS se pueden utilizar para crear redes descentralizadas para almacenar, compartir y acceder a contenido direccionado utilizando identificadores de contenido opacos. Sin embargo, Iroh elimina algunas de las limitaciones de las implementaciones de IPFS como no tener un tamaño máximo de bloque y proporcionar un mecanismo de sincronización de datos a través de la reconciliación de conjuntos basada en rangos sobre

documentos. El roadmap del proyecto incluye llevar la tecnología al navegador a través de WASM, lo que plantea algunas posibilidades interesantes para incorporar la descentralización en aplicaciones web. Si no se desea alojar propios nodos de Iroh, se puede utilizar su servicio en la nube, [iroh.network](#). Ya hay varios SDKs disponibles en una variedad de lenguajes, y uno de los objetivos es ser más amigable para el usuario y más fácil de usar que los sistemas IPFS alternativos. Aunque Iroh aún está en sus primeras etapas, vale la pena seguirlo de cerca, ya que podría convertirse en un actor importante en el espacio de almacenamiento descentralizado.

## 37. Plataformas de modelos de visión de gran tamaño (LVM)

### Evaluar

Los modelos de lenguaje de gran tamaño (LLMs, por sus siglas en inglés), han captado tanta de nuestra atención, que tendemos a pasar por alto los avances en los modelos de visión de gran tamaño (LVMs). Estos modelos pueden ser usados para segmentar, sintetizar, reconstruir y analizar videos e imágenes, a veces en combinación con modelos de difusión o redes neuronales convolucionales estándar. A pesar del potencial de las LVMs para revolucionar la manera que trabajamos con datos visuales, aún nos enfrentamos a retos significativos al adaptarlos y aplicarlos en ambientes de producción. Los datos de video, por ejemplo, presentan retos de ingeniería únicos para recolectar datos de entrenamiento, segmentar y etiquetar objetos, refinar modelos y luego desplegar los modelos resultantes y monitorearlos en producción. Así que mientras los LLMs se prestan a simples interfaces chat o APIs de texto plano, un ingeniero de visión computarizada o ingeniero de datos debe manejar, versionar, anotar y analizar grandes cantidades de datos de video; este trabajo requiere de una interfaz visual. Las plataformas LVM son una nueva categoría de herramientas y servicios - incluyendo [V7](#), [Nvidia Deepstream SDK](#) y [Roboflow](#) — que surgen para atender estos retos. Deepstream y Roboflow son particularmente interesantes para nosotros, dado que combinan un ambiente de desarrollo con interfaz de usuario gráfica integrada para el manejo y creación de anotaciones en video con un conjunto de APIs REST, de Python o C++ para invocar los modelos desde el código de la aplicación.

## 38. OpenBCI Galea

### Evaluar

Hay un creciente interés en el uso de interfaces cerebro-computadora (BCIs) y su potencial aplicación en tecnologías de asistencia. Las tecnologías no invasivas que utilizan electroencefalografía (EEG) y otras señales electrofísicas ofrecen una alternativa de menor riesgo a los implantes cerebrales para aquellos que se están recuperando de lesiones. Ahora están surgiendo plataformas sobre las cuales investigadores y emprendedores pueden desarrollar aplicaciones innovadoras sin tener que preocuparse por los desafíos de procesamiento e integración de señales a bajo nivel. Ejemplos de tales plataformas son [Emotive](#) y [OpenBCI](#) que ofrecen hardware y software de código abierto para desarrollar aplicaciones BCI. El último producto de OpenBCI, el [OpenBCI Galea](#), combina BCI con las capacidades de un auricular de realidad virtual. Ofrece a los desarrolladores acceso a una variedad de flujos de datos fisiológicos sincronizados en el tiempo, junto con sensores de posicionamiento espacial y seguimiento ocular. Esta amplia gama de datos de sensores se puede utilizar para controlar una variedad de dispositivos físicos y digitales. El SDK es compatible con varios lenguajes y hace que los datos de los sensores estén disponibles en [Unity](#) o [Unreal](#). Estamos entusiasmados de ver esta capacidad ofrecida en una plataforma de código abierto, brindando a los investigadores acceso a las herramientas y datos que necesitan para innovar en este campo.

## 39. PGLite

### *Evaluar*

PGLite es una compilación en WASM de una base de datos PostgreSQL. A diferencia de intentos anteriores que requerían una máquina virtual Linux, PGLite construye directamente PostgreSQL en WASM, lo que permite ejecutarlo completamente en el navegador web. Se puede crear una base de datos efímera en la memoria o persistir en el disco mediante indexedDB. Desde la última vez que mencionamos local-first applications en el Radar, las herramientas han evolucionado considerablemente. Con Electric y PGLite, ahora se puede crear aplicaciones locales y reactivas en PostgreSQL.

## 40. SpinKube

### *Evaluar*

SpinKube es un entorno de ejecución serverless de código abierto para WebAssembly en Kubernetes. Mientras Kubernetes ofrece capacidades de auto escalado robustas, el tiempo de arranque en frío de los contenedores aún puede requerir un aprovisionamiento previo para cargas más elevadas. Creemos que el tiempo de inicio de milisegundos de WebAssembly proporciona una solución serverless más dinámica y flexible para cargas de trabajo bajo demanda. Desde nuestra discusión anterior de Spin, el ecosistema WebAssembly ha hecho avances significativos. Nos entusiasma destacar SpinKube, una plataforma que simplifica el desarrollo y despliegue de cargas de trabajo basadas en WebAssembly en Kubernetes.

## 41. Unblocked

### *Evaluar*

Unblocked ofrece descubrimientos de los ciclos de vida del desarrollo del software (SDLC) y de artefactos. Se integra con herramientas comunes de gestión del ciclo de vida de las aplicaciones (ALM) y con herramientas de colaboración para ayudar a los equipos a comprender las bases de código y los recursos relacionados. Mejora la comprensión del código al brindar un contexto relevante e inmediato sobre el código, lo que facilita la navegación y la comprensión de sistemas complejos. Los equipos de ingeniería pueden acceder de manera segura y conforme a las normas a debates, activos y documentos relacionados con su trabajo. Unblocked también captura y comparte el conocimiento local que a menudo reside en miembros experimentados del equipo, lo que hace que la información valiosa sea accesible para todos, independientemente del nivel de experiencia

# Herramientas

## Adoptar

- 42. Bruno
- 43. K9s
- 44. SOPS
- 45. Herramientas para pruebas de regresión visual
- 46. Wiz

## Probar

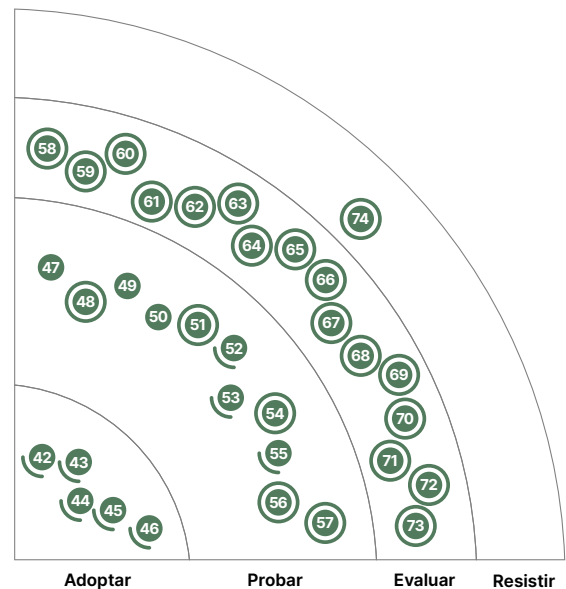
- 47. AWS Control Tower
- 48. CCMenu
- 49. ClickHouse
- 50. Devbox
- 51. Diffastic
- 52. LinearB
- 53. pgvector
- 54. Herramienta de desarrollo de Snapcraft
- 55. Spinnaker
- 56. TypeScript OpenAPI
- 57. Unleash

## Evaluar

- 58. Astronomer Cosmos
- 59. ColPali
- 60. Cursor
- 61. Data Mesh Manager
- 62. GitButler
- 63. Asistente de IA de JetBrains
- 64. Mise
- 65. Mockoon
- 66. Raycast
- 67. ReadySet
- 68. Rspack
- 69. Semantic Router
- 70. Asistente de Ingeniería de Software
- 71. uv
- 72. Warp
- 73. Zed

## Resistir

- 74. CocoaPods



- Nuevo
- Desplazado dentro/afuera
- Ningún cambio

## 42. Bruno

### Adoptar

Bruno es una alternativa de escritorio de código abierto para Postman e Insomnia para pruebas, desarrollo y depuración de APIs. Su objetivo es proveer colaboración, privacidad y seguridad superior con su diseño simple y sin conexión. Las colecciones se almacenan directamente en el sistema de archivos local – escritas un lenguaje de marcado de texto plano, Bru Lang – y puede ser compartido con git u otra herramienta de control de versiones que elija para colaborar. Bruno está disponible tanto en aplicación de escritorio como en herramienta CLI. También ofrece una extensión oficial de VS Code, con planes para soporte de IDE adicional. Bruno se ha convertido en la opción predeterminada para varios equipos en Thoughtworks, pero también recomendamos a los equipos tener precaución al trabajar en entornos VPN y proxy, ya que las solicitudes realizadas en tales condiciones se ha informado que fallan inesperadamente.

## 43. K9s

### Adoptar

K9s ha mejorado sus características de visualización al integrar gráficas y vistas más detalladas. Ofrece una mejor visualización de logs y métricas además ser más flexible en cómo muestra las definiciones de recursos personalizados (CRDs). Las operaciones sobre los pods han sido expandidas también y ahora incluye una mejor integración con herramientas de depuración (p. ej., kubectl debug) y soporte mejorado para ambientes multi-cluster. El soporte para CRDs ha mejorado significativamente al proveer una mejor navegación y administración de estos recursos además de una interacción más fluida con recursos personalizados. El panel de accesos directos ha sido también mejorado para hacerlo más accesible a los desarrolladores que tienen menos experiencia con kubectl. Ésta es una mejora significativa ya que al principio K9s estaba enfocada principalmente en equipos DevOps.

## 44. SOPS

### Adoptar

SOPS es un editor de archivos encriptados que admite varios formatos de encriptación con KMS. Nuestro consejo sobre la gestión de secretos siempre ha sido mantenerlos fuera del código. No obstante, al tener que elegir entre la automatización completa (apegándonos al principio de infraestructura como código) y algunos pasos manuales (utilizando herramientas de tipo Vault) para la administración, aprovisionamiento y rotación de secretos iniciales, los equipos frecuentemente se encuentran frente un dilema. Por ejemplo, nuestros equipos utilizan SOPS para gestionar las credenciales de inicialización en el aprovisionamiento de infraestructura. Sin embargo, hay situaciones en las que es imposible eliminar secretos de los repositorios con código heredado. En esos casos, recurrimos a SOPS para encriptar secretos en archivos de texto. SOPS se integra con almacenes de claves gestionados en la nube, como AWS y GCP Key Management Service (KMS) o Azure Key Vault como fuente de las claves de encriptación. También funciona de manera multiplataforma y admite claves PGP. Muchos de nuestros equipos optan por SOPS como su primera opción para gestionar secretos en el repositorio.

## 45. Herramientas para pruebas de regresión visual

### Adoptar

Hemos destacado las herramientas para pruebas de regresión visual antes y hemos observado sus algoritmos evolucionar desde una comparación primitiva a nivel de píxel hasta ser un sofisticado sistema de coincidencia de patrones y reconocimiento óptico de caracteres (OCR, por sus siglas en inglés). Las herramientas tempranas de regresión visual generaban muchos falsos positivos y eran útiles solo en etapas posteriores de desarrollo cuando la interfaz se volvía estable. [BackstopJS](#) prevenía este problema al configurar selectores y ventanas gráficas para identificar pruebas visuales de elementos específicos en la pantalla. Pero el machine learning ha facilitado la detección y comparación de elementos visuales de manera más precisa, a pesar de que estos se muevan o abarquen contenido dinámico. Estas herramientas de pruebas se han vuelto más útiles y están bien posicionadas para aprovechar los últimos avances en inteligencia artificial y machine learning. Varias herramientas comerciales como [Applitools](#) y [Percy](#), ahora afirman usar IA en sus pruebas de regresión visual. Uno de nuestros equipos ha utilizado Applitools Eyes extensamente y están contentos con los resultados. Aunque las pruebas de regresión visual no son un sustituto para unas pruebas funcionales end-to-end bien escritas, sí son una adición para la caja de herramientas de pruebas. Estamos promoviendo su adopción porque se han convertido en una opción predeterminada segura como elemento de una estrategia integral de pruebas de IU.

## 46. Wiz

### Adoptar

[Wiz](#) se ha convertido en la plataforma de seguridad en la nube preferida en muchos de nuestros proyectos. Nuestros equipos valoran su capacidad para detectar riesgos y amenazas más temprano que otras herramientas similares, debido a que realiza escaneos continuos en busca de cambios. Wiz puede detectar y alertar sobre configuraciones incorrectas, vulnerabilidades y secretos filtrados, tanto en artefactos que aún no se han implementado en entornos de producción (imágenes de contenedores, código de infraestructura) como en cargas de trabajo en vivo (contenedores, máquinas virtuales y servicios en la nube). También apreciamos su potente capacidad de generación de informes tanto para equipos de desarrollo como líderes. Este análisis nos ayuda a comprender cómo una vulnerabilidad puede afectar un servicio determinado, para que podamos resolver los problemas en ese contexto.

## 47. AWS Control Tower

### Probar

[AWS Control Tower](#) sigue siendo nuestra opción preferida para administrar cuentas de AWS en ambientes con múltiples equipos. Proporciona un mecanismo conveniente para pre-configurar controles de seguridad y cumplimiento que se aplicarán automáticamente en nuevas landing zones. Éste es un ejemplo de “[cumplimiento en el punto de cambio](#)” porque los controles son aplicados y verificados cuando una nueva infraestructura es creada, eliminando la necesidad de verificar el cumplimiento de forma manual más tarde. [AWS Control Tower Account Factory for Terraform \(AFT\)](#) ha seguido evolucionando desde nuestro último volumen y ahora está disponible en más regiones de AWS. AFT permite que las cuentas Control Tower sean aprovisionadas mediante un pipeline usando [infraestructura como código](#). Nos agrada que AFT se puede personalizar para enviar webhooks o tomar acciones específicas para integrarse de forma segura con herramientas externas como [GitHub Actions](#). Nuestros equipos han reportado muy buenos resultados usando AWS Control Tower para el manejo de cuentas, pero deseamos que AWS acepte contribuciones de la comunidad al proyecto cuando veamos oportunidades de mejora.

## 48. CCMenu

### Probar

Para los equipos que implementan integración continua, es importante conocer el estado de las builds en el sistema de integración continua (CI). Antes de la pandemia, se mostraba esta información a simple vista en dashboards en grandes pantallas de TV en las salas de equipo. Dado que el trabajo remoto está aquí para quedarse, es necesaria una solución que funcione en las máquinas locales de los desarrolladores. Para la Mac, ese nicho está cubierto por CCMenu, una pequeña aplicación escrita por un Thoughtworker. Originalmente, parte de CruiseControl, funciona con todos los servidores que pueden proporcionar información en formato cctray incluyendo Jenkins y TeamCity. Una actualización reciente ha añadido soporte para GitHub Actions y ha preparado el camino para extender la integración con más servidores de integración continua y métodos de autenticación.

## 49. ClickHouse

### Probar

ClickHouse es una base de datos de procesamiento analítico en línea (OLAP, por sus siglas en inglés) de código abierto y orientada a columnas, diseñada para análisis en tiempo real. Comenzó como un proyecto experimental en 2009 y desde entonces ha madurado hasta convertirse en una base de datos analítica de alto rendimiento y escalable linealmente. Su eficiente motor de procesamiento de consultas junto a la compresión de datos la hacen adecuada para ejecutar consultas interactivas sin pre agregación. ClickHouse también es una excelente opción para almacenar datos de OpenTelemetry. Su integración con Jaeger te permite almacenar grandes volúmenes de trazas y analizarlas de manera eficiente.

## 50. Devbox

### Probar

pesar de los avances en las herramientas de desarrollo, mantener entornos de desarrollo locales consistentes sigue siendo un desafío para muchos equipos. La incorporación de nuevos ingenieros a menudo implica ejecutar comandos o scripts personalizados que pueden fallar de manera impredecible en diferentes máquinas y generar inconsistencias. Para resolver este desafío, nuestros equipos han recurrido cada vez más a Devbox. Devbox es una herramienta de la línea de comandos que proporciona una interfaz accesible para crear entornos de desarrollo locales reproducibles por proyecto, aprovechando el administrador de paquetes Nix sin usar máquinas virtuales o contenedores. Ha agilizado notablemente el flujo de trabajo de incorporación porque, una vez que se ha configurado para una base de código, se necesita un solo comando CLI (devbox shell) para reproducir el entorno definido en un nuevo dispositivo. Devbox es compatible con shell hooks, scripts personalizados y generación de devcontainer.json para la integración con VSCode.

## 51. Diffstastic

### Probar

Diffstastic es una herramienta que sirve para identificar las diferencias de código entre distintos archivos teniendo en cuenta su sintaxis. Esto es muy diferente de las herramientas de comparación de texto, como el venerable comando de Unix diff. Por ejemplo, Diffstastic ignora las líneas insertadas para separar declaraciones largas en lenguajes como Java o TypeScript, las cuáles están delimitadas por punto y coma. La herramienta sólo resalta los cambios que afectan a la sintaxis del código. Para



ello, primero analiza los archivos en árboles sintácticos abstractos y después calcula la distancia entre ellos mediante el algoritmo de Dijkstra. Hemos comprobado que Diffastic es especialmente útil para comprender los cambios cuando se revisan grandes bases de código. Diffastic es compatible con cualquier lenguaje de programación que pueda ser analizado sintácticamente y es compatible con más de 50 lenguajes de programación y formatos de texto estructurado como CSS y HTML. No se trata de una herramienta nueva, pero creemos que merece la pena ser destacada en la era de los asistentes de código basados en LLMs, donde las revisiones manuales de bases de código son cada vez más grandes y críticas.

## 52. LinearB

### *Probar*

LinearB, es una plataforma de inteligencia de ingeniería de software, que ha brindado a nuestros líderes de ingeniería información basada en datos para respaldar la mejora continua. Alinea áreas clave como benchmarking, automatización de flujos de trabajo y las inversiones específicas para mejorar la experiencia y la productividad de los desarrolladores. En nuestra experiencia con LinearB destacamos su capacidad para fomentar una cultura de mejora y eficiencia dentro de los equipos de ingeniería. Nuestros equipos han utilizado la plataforma para realizar un seguimiento de métricas claves de ingeniería, identificar áreas de mejora e implementar acciones basadas en evidencia. Estas capacidades se alinean bien con la propuesta de valor central de LinearB: benchmarking, automatización de la recopilación de métricas y habilitación de mejoras basadas en datos. LinearB se integra con el código fuente, el ciclo de vida de las aplicaciones, CI/CD y herramientas de comunicación y utiliza métricas de ingeniería personalizadas y preconfiguradas para proporcionar información cuantitativa integral sobre la experiencia de desarrollo, la productividad y el desempeño del equipo. Como defensores de DORA, apreciamos el fuerte énfasis de LinearB en estas métricas específicas y su capacidad para medir aspectos clave del rendimiento de la entrega de software, que son esenciales para mejorar la eficiencia. Históricamente, los equipos han enfrentado desafíos al recopilar métricas específicas de DORA, a menudo confiando en paneles personalizados complejos o procesos manuales. LinearB continúa ofreciendo una solución convincente que automatiza el seguimiento de estas métricas y ofrece datos en tiempo real que respaldan la toma de decisiones proactiva en torno a experiencia de desarrollo, productividad y predictibilidad.

## 53. pgvector

### *Probar*

pgvector es una extensión de búsqueda de similitud de vectores de código abierto para PostgreSQL, que permite el almacenamiento de vectores junto con datos estructurados en una única base de datos bien establecida. Aunque carece de algunas características avanzadas de las bases de datos vectoriales especializadas, se beneficia de la compatibilidad con ACID, la recuperación puntual y otras características robustas de PostgreSQL. Con el auge de las aplicaciones generativas impulsadas por IA, vemos un patrón creciente de almacenamiento y búsqueda eficiente de similitudes en vectores incrustados, que pgvector aborda eficazmente. Con el creciente uso de pgvector en entornos de producción, especialmente donde los equipos ya están utilizando un proveedor en la nube que ofrece PostgreSQL gestionado, y su capacidad demostrada para satisfacer las necesidades comunes de búsqueda de vectores sin necesidad de un almacén de vectores independiente, confiamos en su potencial. Nuestros equipos lo han encontrado valioso en proyectos que comparan datos estructurados y no estructurados, demostrando su potencial para una adopción más amplia, y por lo tanto lo estamos moviendo al anillo de prueba.

## 54. Herramienta de desarrollo de Snapcraft

### Probar

Snapcraft es una herramienta de línea de comandos de código abierto para desarrollar y empaquetar aplicaciones autónomas llamadas snaps en Ubuntu, otras distribuciones de Linux y macOS. Los snaps son fáciles de desplegar y mantener en varias plataformas de hardware, incluyendo máquinas Linux, entornos virtuales y sistemas informáticos a bordo de vehículos. Aunque Snapcraft ofrece una tienda de aplicaciones pública para la publicación de snaps, nuestros equipos utilizan la herramienta de desarrollo para empaquetar el sistema de conducción autónoma como un snap sin publicarlo en la tienda de aplicaciones pública. Esto nos permite desarrollar, probar y depurar el sistema de software embebido localmente, mientras se publica en un repositorio de artefactos interno.

## 55. Spinnaker

### Probar

Spinnaker es una plataforma de código abierto para entrega continua creada por Netflix. En esta se implementan manejo de clúster y despliegue de baked-images a la nube como funciones de primera clase. Nos gusta el acercamiento dogmático de Spinnaker para el despliegue de microservicios. En ediciones previas, notamos la falta de capacidad de configurar pipelines como código, pero esto ha sido abordado con la adición del spin CLI. A pesar de que no recomendamos Spinnaker para escenarios de despliegues continuos simples, se ha vuelto la herramienta a elegir por muchos en situaciones complejas con pipelines de despliegue igualmente complejos.

## 56. TypeScript OpenAPI

### Probar

TypeScript OpenAPI (o tsoa) es una alternativa a Swagger para generar especificaciones de OpenAPI para tu código. Es code-first, con los controladores y modelos de TypeScript como la única fuente de verdad, y utiliza las anotaciones o decoradores de TypeScript en vez de los archivos y las configuraciones más complejas necesarias para utilizar OpenAPI tooling para TypeScript. Genera tanto especificaciones 2.0 como 3.0 de la API, y las rutas se pueden generar para Express, Hapi y Koa. Si estás escribiendo APIs en TypeScript, vale la pena echar un vistazo a este proyecto.

## 57. Unleash

### Probar

Aunque usar el feature toggle más simple posible sigue siendo nuestro enfoque recomendado, el crecimiento de equipos y realizar el desarrollo más rápido hacen que gestionar feature toggles a mano sea más complejo. Unleash es una opción ampliamente usada por nuestros equipos para hacer frente a esta complejidad y habilitar flujos CI/CD. Puede ser usado como servicio o self-hosted. Provee SDKs en varios lenguajes otorgando buena experiencia de desarrollador y una interfaz de usuario amigable para su administración. Aunque aún no existe soporte oficial para la especificación OpenFeature se pueden encontrar proveedores para Go y Java, mantenidos por la comunidad. Unleash puede ser usado para simples feature toggles, así como para segmentación y despliegues graduales, lo que lo convierte en una opción adecuada para la gestión de funcionalidad a escala.

## 58. Astronomer Cosmos

### *Evaluar*

Astronomer Cosmos es un plugin de Airflow diseñado para proporcionar un soporte nativo a los flujos de trabajo centrales de dbt en Airflow. Con el plugin instalado, cuando se utiliza DbtDag para envolver un flujo de trabajo de dbt, convierte los nodos de dbt en grupos de tareas/tarea de Airflow, permitiendo que los ingenieros e ingenieras visualicen grafos de dependencias de dbt y el progreso en su ejecución directamente en la interfaz de usuario de Airflow. También da soporte para utilizar conexiones de Airflow en lugar de perfiles de dbt, reduciendo potencialmente la dispersión de configuraciones. Estamos experimentando con la herramienta por su potencial para hacer que el trabajo con dbt en Airflow sea más fluido.

## 59. ColPali

### *Evaluar*

ColPali es una herramienta emergente para la recuperación de documentos PDF que utiliza modelos de lenguaje visual, abordando los desafíos de construir una aplicación robusta de generación mejorada por recuperación (RAG) que pueda extraer datos de documentos multimedia que contengan imágenes, diagramas y tablas. A diferencia de los métodos tradicionales que se basan en embeddings basados en texto o técnicas de reconocimiento óptico de caracteres (OCR), ColPali procesa páginas completas en PDF, utilizando un transformador visual para crear embeddings que consideran tanto el contenido textual como visual. Este enfoque holístico permite una mejor recuperación y una mayor capacidad de razonamiento sobre por qué se recuperan ciertos documentos, mejorando significativamente el rendimiento de RAG en PDFs ricos en datos. Hemos probado ColPali con varios clientes en los que ha mostrado resultados prometedores, aunque la tecnología aún está en sus primeras etapas. Vale la pena evaluarlo, especialmente para organizaciones con datos en documentos visuales complejos.

## 60. Cursor

### *Evaluar*

La carrera por desarrollar herramientas de programación asistidas por inteligencia artificial (AI) está en marcha, y la más llamativa es Cursor. Es un editor de código AI-first diseñado para mejorar la productividad de los desarrolladores mediante la integración profunda de la IA en el flujo de trabajo de codificación. Le hemos prestado atención desde anteriores evaluaciones de radares, pero está claro que la reciente mejora continua de Cursor ha marcado el comienzo de un cambio cualitativo. En nuestro caso, ha demostrado fuertes capacidades de razonamiento contextual en función del código base existente. Mientras que otras herramientas de código de IA como GitHub Copilot tienden a generar y colaborar en torno a fragmentos de código, las operaciones de edición multilínea y multiarchivo de Cursor lo hacen destacar. Al basarse en VSCode ofrece un método de interacción rápido e intuitivo que se ajusta a la lógica del desarrollador. Poderosas operaciones pueden ser completadas con ctrl/cmd+K y ctrl/cmd+L. Cursor está liderando una nueva ronda de competición en herramientas de programación de IA, especialmente en lo que respecta a la interacción del desarrollador y la comprensión del código base.

## 61. Data Mesh Manager

### *Evaluar*

Data Mesh Manager proporciona la capa de metadatos de una típica plataforma de data mesh. En particular, se centra en la definición de productos de datos y la especificación de contratos de datos usando la iniciativa OpenContract y puede integrarse en pipelines de construcción usando la DataContract CLI, asociada. La aplicación también provee un catálogo de datos para descubrir y explorar productos de datos y su metadata, y permite un gobierno federado incluyendo la definición de métricas de calidad de datos y la administración de reglas de calidad de datos. Es una de las primeras herramientas nativas en este espacio lo que significa que no está solamente tratando de adaptar plataformas existentes al paradigma de data mesh.

## 62. GitButler

### *Evaluar*

A pesar de su potencia y utilidad, la interfaz de línea de comandos de Git's es notoriamente compleja cuando se trata de gestionar múltiples ramas y preparar commits dentro de ellas. GitButler es un cliente de Git que proporciona una interfaz gráfica que busca simplificar este proceso. Lo hace rastreando los cambios de archivos no confirmados de forma independiente a Git y luego los organiza en ramas virtuales. Se podría argumentar que esto es una solución para un problema que no debería existir en primer lugar; si haces cambios pequeños y haces push al trunk con frecuencia, no hay necesidad de múltiples ramas. Sin embargo, cuando tu flujo de trabajo incluye pull requests, la estructura de ramas puede volverse compleja, especialmente si hay un ciclo de revisión largo antes de que se pueda fusionar un pull request. Para abordar esto, GitButler también se integra con GitHub, lo que te permite agrupar selectivamente cambios en pull requests y emitirlos directamente desde la herramienta. GitButler es una entrada más en la creciente categoría de herramientas enfocadas en gestionar la complejidad inherente al proceso de pull requests.

## 63. Asistente de IA de JetBrains

### *Evaluar*

El Asistente de IA de JetBrains es un asistente de codificación diseñado para integrarse sin problemas con todos los IDE de JetBrains apoyando el autocompletado de código, la generación de pruebas y la adherencia a de la guía de estilo. Desarrollado sobre modelos como OpenAI y Google Gemini, se destaca por su capacidad para garantizar un resultado consistente al recordar estilos de codificación para futuras sesiones. Nuestros desarrolladores encontraron sus capacidades de generación de pruebas particularmente útiles y notaron su habilidad para manejar resultados más largos sin problemas de estabilidad. Sin embargo, a diferencia de algunos competidores, JetBrains no aloja sus propios modelos, lo que puede no funcionar para clientes preocupados por el manejo de datos hacia terceros. Aun así, la integración de la herramienta con los IDEs de JetBrains la convierte en una opción prometedora para los equipos que exploran asistentes de codificación impulsados por IA.

## 64. Mise

### *Evaluar*

Los desarrolladores que trabajan en entornos políglotas, se encuentran con frecuencia con múltiples versiones de diferentes lenguajes y herramientas. mise tiene como objetivo resolver ese problema proporcionando una herramienta que reemplaza nvm, pyenv, rbenv and rustup, entre otras, y es un reemplazo directo para asdf. Mise está escrito en Rust para tener una interacción rápida con el terminal, y a diferencia de asdf, el cual usa paquetes basados en terminal, Mise modifica la variable

de entorno PATH por adelantado, lo que hace que las ejecuciones se llamen directamente. Esto es en parte porque Mise es más rápido que asdf. Para aquellos desarrolladores que ya están familiarizados con asdf, Mise proporciona la misma funcionalidad, pero con unas pocas diferencias clave. Estando escrito en Rust, es más rápido y tiene algunas características que asdf no tiene, como por ejemplo, la habilidad de instalar múltiples versiones de la misma herramienta al mismo tiempo y ser capaz de recordar comandos, incluyendo coincidencia difusa. También proporciona un ejecutor de tareas integrado, útil para cosas como linters, tests, compiladores, servidores y otras tareas que son específicas de un proyecto. Si estás un poco harta de tener que usar múltiples herramientas para administrar tu entorno de desarrollo, y de la sintaxis, algunas veces incómoda de otras herramientas, Mise definitivamente te merecerá la pena.

## 65. Mockoon

### *Evaluar*

Mockoon es una herramienta de código abierto para la simulación de APIs. Ofrece una interfaz intuitiva, rutas personalizables y respuestas dinámicas, además de la capacidad de automatizar la creación de conjuntos de datos simulados. Mockoon es compatible con OpenAPI y te permite generar diferentes escenarios que pueden probarse localmente e integrarse en un pipeline de desarrollo. También puedes crear simulaciones parciales (partial mocks) interceptando las solicitudes y simulando solo las llamadas que están definidas en Mockoon. Estas simulaciones parciales ayudan a emular rutas o endpoints específicos de una API, mientras que el resto de solicitudes se redirigen a servidores reales. Aunque las simulaciones parciales pueden ser útiles en ciertos escenarios, existe el riesgo de un uso excesivo, lo que podría añadir complejidad innecesaria. Aparte de esto, Mockoon sigue siendo una herramienta valiosa para configurar rápidamente APIs simuladas y mejorar y automatizar los flujos de trabajo de desarrollo.

## 66. Raycast

### *Evaluar*

Raycast es un launcher freemium para macOS que te permite abrir aplicaciones rápidamente, ejecutar comandos, buscar archivos y automatizar tareas desde el teclado. Nuestros equipos valoran sus funciones predeterminadas para desarrolladores y su fácil extensibilidad, que permite interactuar con aplicaciones y servicios de terceros como VSCode, Slack, Jira y Google, entre otros muchos. Raycast está diseñada para mejorar la productividad y minimizar los cambios de contexto, lo que la convierte en una herramienta útil para quienes buscan optimizar sus tareas diarias. Los usuarios Pro tienen acceso a Raycast AI, un asistente de búsqueda especializado impulsado por inteligencia artificial.

## 67. ReadySet

### *Evaluar*

ReadySet es un caché de consultas para MySQL y PostgreSQL. A diferencia de soluciones de caching que dependen de invalidación manual, ReadySet aprovecha los flujos de replicación de las bases de datos para actualizar su caché incrementalmente. A través de vistas materializadas parciales, ReadySet alcanza latencias de cola más bajas que una réplica de lectura tradicional. ReadySet es compatible con MySQL y PostgreSQL, por lo que puedes desplegarlo frente a tu base de datos para escalar horizontalmente cargas de trabajo sin requerir cambios en la aplicación.

## 68. Rspack

### *Evaluar*

Muchos de nuestros equipos que trabajan en frontends basados en la web se han movido de herramientas de empaquetamiento más antiguas — como [Webpack](#) — a [Vite](#). Un nuevo participante en este campo es [Rspack](#), que después de 1.5 años en desarrollo ha visto su [release 1.0](#). Diseñado como un reemplazo directo para Webpack, es compatible con los plugins y loaders del ecosistema de Webpack. Esto puede ser una ventaja por sobre Vite al momento de migrar configuraciones complejas de Webpack. La razón principal por la que nuestros equipos están migrando a nuevas herramientas como Vite y Redpack es la experiencia de desarrollo, y en particular, la velocidad. Nada rompe más el flujo de desarrollo que tener que esperar un minuto o dos antes de conseguir retroalimentación de los últimos cambios en el código. Escrita en Rust, Rspack entrega un rendimiento significativamente más rápido que Webpack, y en muchos casos, es incluso más rápido que Vite.

## 69. Semantic Router

### *Evaluar*

Cuando se construyen aplicaciones basadas en LLM, es fundamental determinar la intención de un usuario antes de enrutar una petición a un agente especializado o invocar un flujo específico. [Semantic Router](#) actúa como una capa superrápida de toma de decisiones para LLMs y agentes, permitiendo un enrutamiento eficiente y fiable de las peticiones basado en el significado semántico. Al utilizar embeddings vectoriales para inferir la intención, Semantic Router reduce las llamadas innecesarias a los LLM, ofreciendo un enfoque más ágil y rentable para comprender la intención del usuario. Su potencial se extiende más allá de la inferencia de intenciones, sirviendo como bloque de construcción versátil para diversas tareas semánticas. La velocidad y flexibilidad que ofrece lo sitúan como un fuerte competidor en entornos que requieren una toma de decisiones rápida y en tiempo real sin la sobrecarga de los LLM.

## 70. Asistente de Ingeniería de Software

### *Evaluar*

Uno de los temas más interesantes en este momento en el espacio GenAI es el concepto de Asistente de Ingeniería de Software. Estas herramientas de asistencia al código hacen más que simplemente ayudar a los ingenieros con fragmentos de código aquí y allá; amplifica el tamaño del problema que pueden resolver, idealmente de forma autónoma y con mínima intervención humana. La idea es que estas herramientas puedan tomar una incidencia de GitHub o un ticket de Jira y proponer un plan y cambios en el código para implementarlo, o incluso crear una revisión de código para que una persona lo revise. Si bien éste es el siguiente paso lógico para aumentar el impacto de la asistencia al código con IA, el publicitado objetivo de asistentes genéricos que puedan cubrir una amplia gama de tareas de programación es muy ambicioso, y el estado actual de las herramientas aún no lo demuestra de manera convincente. Sin embargo, podemos ver que esto funcionará más pronto que tarde para un alcance más limitado de tareas sencillas, lo que liberará tiempo de los desarrolladores para trabajar en problemas más complejos. Las herramientas que están lanzando y comercializando versiones beta de agentes incluyen [GitHub Copilot Workspace](#), [qodo flow](#), [Agentes para jira JIRA](#) de Tabnine o [Amazon Q Developer](#). La referente [SWE Bench](#) enumera más herramientas en ese espacio, pero recomendamos tomar los benchmark en el espacio de la IA con cautela..

## 71. uv

### *Evaluar*

Rust es muy adecuado para escribir herramientas de línea de comandos debido a su rápido rendimiento de arranque, y vemos gente reescribiendo algunas cadenas de herramientas en él. En el anterior Tech Radar mencionamos Ruff, un linter para Python escrito en Rust. En esta edición, evaluamos uv, una herramienta de gestión de paquetes de Python escrita en Rust. La propuesta de valor de uv es ser ultrarrápida y supera a otras herramientas de gestión de paquetes de Python por un amplio margen en sus benchmarks. Sin embargo, durante la evaluación para este radar, analizamos si optimizar segundos para las herramientas de compilación es realmente una mejora. En comparación con el rendimiento, lo más importante para un sistema de gestión de paquetes es el ecosistema, la madurez de la comunidad y el soporte a largo plazo. Dicho esto, el feedback del equipo del proyecto nos ha demostrado que esta mejora en el margen de la velocidad podría ser una gran ventaja para mejorar los ciclos de feedback y la experiencia general de desarrollo: tendemos a hacer que el almacenamiento en caché de la CI/CD sea muy complejo de forma manual para lograr esta pequeña mejora del rendimiento. uv simplifica la gestión de nuestro entorno Python. Teniendo en cuenta que todavía hay mucho margen de mejora en la gestión de paquetes y entornos para desarrollo en Python, creemos que uv es una opción que vale la pena evaluar.

## 72. Warp

### *Evaluar*

Warp es un terminal para macOS y Linux. Divide las salidas de comandos en bloques para mejorar la legibilidad. Warp cuenta con capacidades impulsadas por IA tales como sugerencias inteligentes de comandos y procesamiento de lenguaje natural. También incluye una funcionalidad de notebooks que permite a los usuarios organizar comandos y salidas, así como agregar anotaciones y documentación. Se puede aprovechar estas funcionalidades para crear archivos README o materiales de inducción y proporcionar una manera estructurada e interactiva de presentar y gestionar flujos de trabajo en el terminal. Warp se integra fácilmente con Starship, un indicador multiplataforma flexible, lo que te permite personalizar la experiencia del terminal y extraer información sobre procesos en ejecución, la versión específica de una herramienta que estés usando, detalles de Git o el usuario actual de Git, entre otros detalles.

## 73. Zed

### *Evaluar*

Después del cierre del proyecto del editor de texto Atom sus creadores construyeron un nuevo editor llamado Zed. Escrito en Rust y optimizado para aprovechar el hardware moderno, Zed se siente rápido. Tiene todas las características que esperamos de un editor moderno: soporte para muchos lenguajes de programación, un terminal incorporado y edición multibuffer, por mencionar algunos. La codificación asistida por IA está disponible mediante la integración con varios proveedores de LLM. Como fervientes practicantes de la programación en pareja, estamos intrigados por la función de collaboration feature integrada en Zed. Los desarrolladores se encuentran a través de sus IDs de GitHub y pueden colaborar en el mismo espacio de trabajo en tiempo real. Es demasiado pronto para saber si los equipos de desarrollo pueden y quieren escapar del atractivo del ecosistema de Visual Studio Code pero Zed es una alternativa a explorar.

## 74. CocoaPods

### *Resistir*

CocoaPods ha sido una herramienta habitual de gestión de dependencias para proyectos en Swift y Objective-C. Sin embargo, el equipo de CocoaPods anunció que el proyecto se encuentra en modo de mantenimiento después de más de una década siendo una herramienta clave para desarrollar en iOS y macOS. Aunque la herramienta y sus recursos seguirán disponibles, no habrá más desarrollo activo. Se recomienda a los equipos de desarrollo cambiar a Swift Package Manager, que ofrece una integración nativa con Xcode y un mejor soporte a largo plazo por parte de Apple.



# Lenguajes y Frameworks

## Adoptar

- 75. dbt
- 76. Testcontainers

## Probar

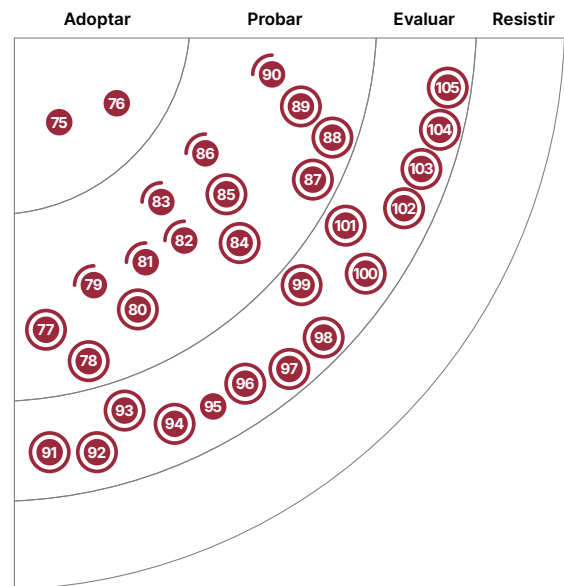
- 77. CAP
- 78. CARLA
- 79. Databricks Asset Bundles
- 80. Instructor
- 81. Kedro
- 82. LiteLLM
- 83. Llamaindex
- 84. LLM Guardrails
- 85. Medusa
- 86. PKI
- 87. ROS 2
- 88. seL4
- 89. SetFit
- 90. vLLM

## Evaluar

- 91. Apache XTable™
- 92. dbldatagen
- 93. DeepEval
- 94. DSPy
- 95. Flutter para la Web
- 96. kotaemon
- 97. Lenis
- 98. LLMingua
- 99. Microsoft Autogen
- 100. Pingora
- 101. Ragas
- 102. Score
- 103. shadcn
- 104. Slint
- 105. SST

## Resistir

—



- Nuevo
- Desplazado dentro/afuera
- Ningún cambio

## 75. dbt

### Adoptar

dbt sigue siendo una opción sólida y sensata para implementar transformaciones de datos en pipelines ELT. Nos gusta que favorece un enfoque riguroso de ingeniería y habilita prácticas como modularidad, capacidad de prueba y reutilización de transformaciones basadas en SQL. dbt se integra bien con muchos data warehouses en la nube, lakehouses y bases de datos, incluidos Snowflake, BigQuery, Redshift, Databricks y Postgres, y cuenta con un ecosistema saludable de paquetes comunitarios. El soporte nativo que se introdujo recientemente (en dbt core 1.8+ y la experiencia recién introducida de dbt Cloudsin versión”) para las pruebas unitarias, refuerza aún más su posición en nuestro conjunto de herramientas. A nuestros equipos les gusta la nueva característica de pruebas unitarias, ya que les permite definir fácilmente datos de prueba estáticos, establecer expectativas de salida y probar tanto los modos de actualización incremental como de actualización completa de sus pipelines. En muchos casos, esto les ha permitido retirar scripts desarrollados internamente, manteniendo el mismo nivel de calidad.

## 76. Testcontainers

### Adoptar

En nuestra experiencia, Testcontainers es una opción predeterminada útil para crear entornos confiables para ejecutar pruebas. Es una librería adaptada a múltiples lenguajes, que “Dockeriza” dependencias de pruebas comunes – incluyendo diferentes tipos de bases de datos, tecnologías de colas, servicios en la nube y dependencias de pruebas de interfaces de usuario, como navegadores web – con la capacidad de ejecutar Dockerfiles personalizados cuando sea necesario. Recientemente, se lanzó una versión de escritorio que permite la gestión visual de sesiones de pruebas y la capacidad de gestionar escenarios más complejos, lo que nuestros equipos han encontrado muy útil.

## 77. CAP

### Probar

CAP es una biblioteca de .NET que implementa el patrón Outbox. Cuando se trabaja con sistemas de mensajes distribuidos como RabbitMQ o Kafka es común encontrar dificultades para asegurar la atomicidad de las actualizaciones de bases de datos y publicaciones de eventos. CAP permite conseguir este objetivo guardando la petición de publicación del evento en la misma transacción de base de datos que causó el evento. Consideramos que CAP es muy útil porque ofrece soporte para varias bases de datos y sistemas de mensajería distribuidos al tiempo que garantiza la entrega de notificaciones al menos una vez.

## 78. CARLA

### Probar

CARLA es un simulador de código abierto para investigación de conducción autónoma, usado para probar este tipo de sistemas antes de su despliegue en producción. Ofrece flexibilidad para crear y reutilizar modelos 3D de vehículos, terrenos, humanos y animales, entre otros, haciendo posible simular escenarios como un peatón entrando en la calzada o encontrarse con un vehículo que viene en sentido contrario a una velocidad determinada. El sistema de conducción autónoma que se está probando debe reconocer esos actores dinámicos y realizar las acciones apropiadas, como frenar. Nuestros equipos usan CARLA para el desarrollo y pruebas continuas de sistemas de conducción autónoma.

## 79. Databricks Asset Bundles

### Probar

Databricks Asset Bundles (DABs), que alcanzó disponibilidad general en abril de 2024, se está convirtiendo en la herramienta predilecta para empaquetar y desplegar recursos de Databricks que facilitan la adopción de técnicas de desarrollo de software en nuestros equipos de datos. DABs soporta el empaquetamiento de la configuración de flujos de trabajo y tareas, al igual que el código a ser ejecutado en dichas tareas, como un paquete que puede ser desplegado a múltiples entornos mediante pipelines de CI/CD. Incluye plantillas para recursos comúnmente utilizados y tiene soporte para plantillas personalizadas, lo que permite la creación de plantillas de servicios a la medida para proyectos de ingeniería de datos y ML. Nuestros equipos están incrementalmente adoptando esta tecnología como una parte clave de sus workflows de ingeniería. A pesar de que DABs incluye plantillas para notebooks y soporta desplegarlos en producción, no recomendamos llevar notebooks a producción. En su lugar impulsamos la creación de código para producción de manera intencional con las prácticas de resiliencia, soporte, escalabilidad y mantenimiento necesarias para este tipo de trabajo.

## 80. Instructor

### Probar

Cuando usamos bots de conversación basados en modelos de lenguaje de gran tamaño (LLM, por sus siglas en inglés) como usuarios finales, generalmente devuelven una respuesta no estructurada en lenguaje natural. Cuando construimos aplicaciones con IA generativa que son algo más que bots de conversación, puede ser útil que el bot devuelva una respuesta estructurada en JSON, YAML u otros formatos para después procesar esta respuesta y utilizarla en la aplicación. Sin embargo, ya que los LLM no son deterministas, puede que no siempre hagan lo que les pedimos. Instructor es una biblioteca que nos ayuda a solicitar salida estructurada a LLMs. Se puede definir la estructura de salida deseada y configurar reintentos si el modelo no devuelve una respuesta en el formato solicitado. Además, ya que la experiencia de usuario óptima con LLMs suele ser transmitir los resultados según se generan en lugar de esperar a tener la respuesta completa, Instructor también se encarga de procesar estructuras parciales a partir de un flujo de datos.

## 81. Kedro

### Probar

Kedro ha mejorado significativamente como herramienta para MLOps y ha mantenido su enfoque en prácticas de modularidad e ingeniería, lo que nos gustaba desde el inicio. Un paso que resalta su modularidad es la introducción del paquete autónomo kedro-datasets que desacopla el código de los datos. Kedro ha añadido mejoras en su CLI, plantillas de inicio de proyecto y capacidades de telemetría. Adicionalmente, el reciente lanzamiento de una extensión en VS Code es un buen impulso para la experiencia del desarrollador.

## 82. LiteLLM

### *Probar*

LiteLLM es una librería para facilitar la integración con varias APIs de proveedores de modelos de lenguaje de gran tamaño (LLM, por sus siglas en inglés de Large Language Model) que estandariza las interacciones mediante un formato de API de OpenAI. Tiene soporte para un amplio número de proveedores y modelos y ofrece una interfaz unificada para completar, hacer embedding y generar imágenes. LiteLLM simplifica la integración al traducir las entradas para que encajen con los requisitos de cada endpoint específico de cada proveedor. También ofrece un framework que es necesario para implementar muchas de las características operacionales que necesita una aplicación en producción, como caching, logging, rate limiting y balanceo de carga. Esto asegura una operación uniforme entre distintos LLMs. Nuestros equipos utilizan LiteLLM para facilitar el cambio entre distintos modelos, algo necesario en el contexto actual donde los modelos evolucionan rápidamente. Es importante tener en cuenta que al hacer esto, las respuestas de los modelos a prompts idénticos varían, lo que indica que un método de invocación consistente por sí solo puede no ser suficiente para optimizar por completo el rendimiento al completar texto. Además, cada modelo implementa funcionalidades add-on de forma única, por lo que una sola interfaz puede no ser suficiente para todos. Por ejemplo, uno de nuestros equipos tuvo dificultades en aprovechar la invocación de funciones a un modelo AWS Bedrock mediante proxying a través de LiteLLM.

## 83. LlamaIndex

### *Probar*

LLamaIndex incluye motores que permiten diseñar aplicaciones LLM específicas de dominio, de contexto aumentado y admite tareas como la ingestión de datos, la indexación por vectores y la respuesta a preguntas en lenguaje natural sobre documentos, por mencionar algunas. Nuestros equipos utilizaron LlamaIndex para construir un pipeline de generación mejorada por recuperación (RAG) que automatiza la ingesta de documentos, indexa las representaciones vectoriales de los documentos y consulta estas representaciones según las entradas del usuario. Al utilizar LlamaHub, tu puedes extender y personalizar los módulos de LlamaIndex para poder ajustarse a tus necesidades y así construir, por ejemplo, las aplicaciones LLM con tus LLMs preferidos, representaciones vectoriales y proveedores de almacenamiento de vectores preferidos.

## 84. LLM Guardrails

### *Probar*

LLM Guardrails son un conjunto de pautas, políticas o filtros diseñados para evitar que los modelos de lenguaje de gran tamaño (LLMs) generen contenido dañino, engañoso o irrelevante. Las barreras también pueden usarse para proteger a las aplicaciones LLM de usuarios malintencionados que intenten abusar del sistema mediante técnicas como la manipulación de inputs. Actúan como una red de seguridad al establecer límites para que el modelo procese y genere contenido. Existen algunos frameworks emergentes en este ámbito como NeMo Guardrails, Guardrails AI y Aporia Guardrails que nuestros equipos han encontrado útiles. Recomendamos que toda aplicación de LLM tenga guardrails implementadas y que sus reglas y políticas se mejoren continuamente. Son cruciales para construir aplicaciones de chat responsables y confiables con LLMs.

## 85. Medusa

### Probar

En nuestra experiencia, la mayoría de soluciones de comercio electrónico para construir páginas web de compras caen en la trampa del 80/20 — podemos construir fácilmente el 80% de lo que queremos pero no podemos hacer nada sobre el 20% restante. Medusa ofrece un buen balance. Es una plataforma de comercio de código abierto altamente personalizable que permite a los desarrolladores crear experiencias de compra únicas y a la medida que pueden ejecutarse localmente o en la plataforma de Medusa. Construido con Next.js y PostgreSQL, Medusa acelera el proceso de desarrollo con su amplio conjunto de módulos — desde un carrito de compras básico y manejo de órdenes hasta funciones avanzadas como módulos de tarjetas de regalo y cálculo de impuestos para diferentes regiones. Hemos encontrado a Medusa como un valioso framework y lo hemos aplicado en algunos proyectos.

## 86. Pkl

### Probar

Pkl es un lenguaje y herramienta de configuración de código abierto, creado inicialmente para uso interno en Apple. Su funcionalidad principal es su tipo y sistema de validación, que permite detectar errores de configuración antes de la ejecución del despliegue. Pkl ha permitido a nuestros equipos reducir la duplicidad de código (en casos como sobrescritura de ambientes) y realizar la validación antes de aplicar los cambios de configuración a los ambientes en vivo. Genera archivos JSON, PLIST, YAML y .properties; y cuenta con una amplia integración de IDE y lenguaje, incluida la generación de código.

## 87. ROS 2

### Probar

ROS 2 Es un framework de código abierto diseñado para el desarrollo de sistemas robóticos. Proporciona un conjunto de librerías y herramientas que permiten la implementación modular de aplicaciones, cubriendo funciones como la comunicación entre procesos, la ejecución multihilo y la calidad de servicio. ROS 2 se basa en su predecesor y ofrece funciones mejoradas en tiempo real, mayor modularidad, mayor compatibilidad con diversas plataformas y sensible defaults. ROS 2 está ganando aceptación en el sector de la industria automotriz; su arquitectura basada en nodos y su modelo de comunicación basado en tópicos resultan especialmente atractivos para fabricantes de aplicaciones complejas y en evolución dentro del vehículo, como la funcionalidad de conducción autónoma.

## 88. seL4

### Probar

En los vehículos definidos por software (SDV) u otros escenarios críticos para la seguridad, la estabilidad en tiempo real del sistema operativo es crucial. Unas pocas empresas monopolizan este campo debido a sus elevadas barreras de entrada, por lo que soluciones de código abierto como seL4 ason muy apreciadas. seL4 es un micronúcleo de sistema operativo de alto rendimiento y garantía. Utiliza métodos de verificación formal para garantizar «matemáticamente» que el comportamiento del sistema operativo se ajusta a la especificación. Su arquitectura de micronúcleo también minimiza las responsabilidades centrales para garantizar la estabilidad del sistema. Hemos visto a empresas de EV como NIO participar en el ecosistema seL4, y es posible que en el futuro se produzcan más desarrollos en este ámbito.

## 89. SetFit

### Probar

La mayoría de las herramientas basadas en IA disponibles actualmente, son generativas — generan textos e imágenes y usan transformers generativos pre-entrenados (GPTs por sus siglas en inglés) para hacerlo. Para casos de uso que requieren trabajar con texto existente — para clasificar fragmentos de texto o determinar intención — los transformers de oraciones son la herramienta a elegir. En este contexto SetFit es un framework para el fine-tuning de transformers de oraciones. Nos gusta SetFit porque usa aprendizaje contrastivo para separar diferentes clases de intención, a menudo logrando una separación clara con un conjunto pequeño de ejemplos, 25 o incluso menos. Los transformers de oraciones también pueden jugar un rol en un sistema de IA generativa. Hemos utilizado con éxito SetFit para la detección de intención en un chatbot de atención al cliente que usa un LLM; y a pesar de que conocemos la API de moderación de OpenAI, hemos elegido un clasificador basado en SetFit para realizar fine-tuning adicional y obtener un filtrado más estricto.

## 90. vLLM

### Probar

vLLM es un motor de inferencia de alto rendimiento y gestión de memoria eficiente para LLM que puede ejecutarse en la nube o en servidores propios. Admite perfectamente múltiples modelos de arquitectura y modelos populares de código abierto. Nuestros equipos despliegan tareas vLLM en plataformas GPU como NVIDIA DGX e Intel HPC, alojando modelos como por ejemplo Llama 3.1(8B and 70B), Mistral 7B y Llama-SQL para la asistencia en desarrollo de código, búsqueda de conocimiento e interacciones de bases de datos en lenguaje natural. vLLM es compatible con el estándar de OpenAI SDK, facilitando un servicio de modelo consistente. El catálogo de modelos de IA de Azure utiliza un contenedor de inferencia personalizado para mejorar el rendimiento del servicio de modelos, con vLLM como motor de inferencia predeterminado debido a su alto rendimiento y gestión eficiente de la memoria. El framework vLLM está emergiendo como el modelo predeterminado de despliegues a larga escala.

## 91. Apache XTable™

### Evaluar

Entre los formatos de tabla abiertos disponibles que soportan lakehouses — como Apache Iceberg, Delta and Hudi — no ha surgido un claro ganador. En cambio, estamos viendo herramientas que permiten la interoperabilidad entre estos formatos. Por ejemplo, Delta UniForm, fsoporta la interoperabilidad unidireccional al permitir que los clientes de Hudi y Iceberg lean tablas de Delta. Otro nuevo participante en este espacio es Apache XTable™, un proyecto incubadora de Apache que facilita la interoperabilidad omnidireccional entre Hudi, Delta y Iceberg. Al igual que UniForm, convierte los metadatos entre estos formatos sin duplicar los datos subyacentes. XTable podría ser útil para equipos que estén experimentando con diferentes formatos de tablas. Sin embargo, para un uso a largo plazo, dada la diferencia de las características entre estos formatos, depender en gran medida de la interoperabilidad omnidireccional podría hacer que los equipos solo puedan utilizar el “mínimo común denominador” de las funcionalidades.

## 92. dbldatagen

### *Evaluar*

Preparar los datos de prueba para ingeniería de datos es un gran desafío. Transferir datos desde producción a ambientes de prueba puede ser riesgoso, por lo que los equipos a menudo optan por utilizar datos falsos o sintéticos en su lugar. En este Radar, exploramos enfoques novedosos como datos sintéticos para pruebas y entrenamiento de modelos. Sin embargo, en muchas ocasiones, la generación procedural de bajo costo es suficiente. dbldatagen (Generador de Datos de Databricks Labs) es una de esas herramientas; se trata de una biblioteca de Python para generar datos sintéticos dentro del entorno de Databricks, utilizada para pruebas, benchmarking, demos y muchos otros usos. dbldatagen puede generar datos sintéticos a gran escala, alcanzando hasta miles de millones de filas en cuestión de minutos, y soporta varios escenarios como múltiples tablas, change data capture y operaciones de merge/join. Maneja bien los tipos primitivos de Spark SQL, genera rangos y valores discretos, y aplica distribuciones específicas. Al crear datos sintéticos utilizando el ecosistema de Databricks, dbldatagen es una opción que vale la pena evaluar.

## 93. DeepEval

### *Evaluar*

DeepEval Es un framework de evaluación de código abierto basado en Python, utilizado para evaluar el rendimiento de los LLM. Puedes usarlo para evaluar la generación aumentada por recuperación (RAG) y otros tipos de aplicaciones creadas con frameworks populares como LlamaIndex o LangChain. También sirve para establecer líneas base y benchmark al comparar diferentes modelos de acuerdo a tus necesidades. DeepEval proporciona un conjunto completo de métricas y funciones para evaluar el rendimiento de los LLM, incluida la detección de alucinaciones, la relevancia de las respuestas y la optimización de hiper parámetros. Ofrece integración con pytest y, junto con sus aserciones, puedes fácilmente integrar el conjunto de pruebas en un pipeline de integración continua (CI). Si trabajas con LLM, considera probar DeepEval para mejorar tu proceso de pruebas y garantizar la fiabilidad de tus aplicaciones.

## 94. DSPy

### *Evaluar*

La mayoría de las aplicaciones basadas en modelos de lenguaje hoy en día confían en plantillas de prompts ajustadas manualmente para tareas específicas. DSPy, un framework para desarrollar tales aplicaciones, toma un enfoque diferente que prescinde de la ingeniería de prompts directa. En su lugar, introduce abstracciones de más alto nivel orientadas en el flujo del programa (a través de módulos que se pueden poner en capas unos encima de otros), métricas que optimizar y datos con los que entrenar/probar. Entonces optimiza los prompts y/o pesos del modelo de lenguaje subyacente basándose en esas métricas que se han definido. El código resultante se parece mucho más al entrenamiento de redes neuronales con PyTorch. Encontramos que el enfoque que toma es estimulante por ser diferente y pensamos que vale la pena experimentar con él.

## 95. Flutter para la Web

### *Evaluar*

Flutter es conocido por su soporte multiplataforma para aplicaciones iOS y Android. Ahora, se ha expandido a más plataformas. Ya hemos evaluado Flutter para la Web : nos permite crear aplicaciones para iOS, Android y el navegador a partir de la misma base de código. No todas las aplicaciones web tienen sentido en Flutter, pero creemos que Flutter es especialmente adecuado para casos como aplicaciones web progresivas, aplicaciones de una sola página y la conversión de aplicaciones móviles Flutter existentes a la web. Flutter ya admitía WebAssembly (WASM) como target de compilación en su canal experimental, lo que significa que estaba en desarrollo activo con posibles errores y problemas de rendimiento. Las versiones más recientes lo han hecho estable. El rendimiento de las aplicaciones web Flutter compiladas con target WASM es muy superior al de su target JavaScript. El rendimiento casi nativo en diferentes plataformas es también la razón por la que muchos desarrolladores eligen inicialmente Flutter.

## 96. kotaemon

### *Evaluar*

kotaemon es una herramienta y framework de código abierto basado en RAG (generación aumentada por recuperación, por sus siglas en inglés de retrieval-augmented generation) para construir aplicaciones de Q&A y documentos de bases de conocimiento. Puede entender varios tipos de documento, incluyendo formatos PDF y DOC, y provee una interfaz web basada en Gradio, la que permite a los usuarios organizar e interactuar con la base de conocimiento a través de una interfaz de chat. Incluye pipelines de RAG con un vector store y puede ser extendido con SDKs. kotaemon también cita los documentos fuente en sus respuestas, junto a vistas previas de web embebidas y un grado de relevancia. Para cualquiera que desee crear una aplicación Q&A con documentos basados en RAG, este framework personalizable es un muy buen punto de partida.

## 97. Lenis

### *Evaluar*

Lenis es una librería ligera y poderosa para obtener un desplazamiento (scrolling) fluido en navegadores modernos. Habilita experiencias como sincronizar el desplazamiento con WebGL o efectos de paralaje. Es ideal para equipos que construyen páginas con interacciones que utilizan desplazamiento fluido. Nuestras desarrolladoras han visto que Lenis ofrece una experiencia transparente y fácil de usar al crear desplazamientos fluidos. De todas maneras, la librería puede tener ciertos problemas con la accesibilidad, particularmente con el desplazamiento horizontal y vertical, lo que puede confundir a usuarios con discapacidades. Aunque es atractivo visualmente, necesita una implementación adecuada para mantener la accesibilidad.

## 98. LLMingua

### *Evaluar*

LLMingua mejora la eficiencia de los LLMs al comprimir las entradas del usuario usando un modelo de lenguaje pequeño para eliminar los tokens no esenciales con una pérdida mínima de rendimiento. Este enfoque permite a los LLMs mantener el raciocinio y aprendizaje dentro del contexto mientras procesan eficientemente entradas más largas, afrontando retos como eficiencia de costos, latencia de inferencia y gestión de contexto. LLMingua es perfecto para optimizar el rendimiento de inferencia de los LLM, ya que es compatible con distintos LLMs, no necesita entrenamiento adicional y soporta frameworks como LLamaIndex.



## 99. Microsoft Autogen

### *Evaluar*

Microsoft Autogen es un framework de código abierto que simplifica la creación y orquestación de agentes de IA, permitiendo la colaboración entre múltiples agentes para resolver tareas complejas. Soporta tanto flujos de trabajo autónomos como aquellos con intervención humana, ofreciendo compatibilidad con una variedad de modelos de lenguaje de gran tamaño (LLMs) y herramientas para la interacción entre agentes. Uno de nuestros equipos utilizó Autogen en un cliente para construir una plataforma impulsada por IA en la que cada agente representaba una habilidad específica, como la generación de código, la revisión de código o el resumen de documentación. El framework permitió al equipo crear nuevos agentes de manera fluida y consistente, definiendo el modelo y el flujo de trabajo adecuados. Utilizaron Llamaindex para orquestar los flujos de trabajo, permitiendo a los agentes gestionar tareas como la búsqueda de productos y sugerencias de código de manera eficiente. Aunque Autogen ha mostrado ser prometedor, especialmente en entornos de producción, persisten preocupaciones sobre la escalabilidad y la gestión de la complejidad a medida que se añaden más agentes. Se necesita una evaluación adicional para determinar su viabilidad a largo plazo en la escalabilidad de sistemas basados en agentes.

## 100. Pingora

### *Evaluar*

Pingora es un framework de Rust para construir servicios de red rápidos, fiables y programables. Originalmente desarrollado por Cloudflare para abordar las deficiencias de Nginx, Pingora ya está mostrando un gran potencial, ya que los nuevos proxies como River se están construyendo sobre sus cimientos. Aunque la mayoría de nosotros no nos enfrentamos al nivel de escala de Cloudflare, nos encontramos con escenarios en los que el enrutamiento flexible de la capa de aplicación es esencial para nuestros servicios de red. La arquitectura de Pingora nos permite aprovechar toda la potencia de Rust en estas situaciones sin sacrificar la seguridad ni el rendimiento.

## 101. Ragas

### *Evaluar*

Ragas es un framework diseñado para evaluar el rendimiento de los pipelines de generación aumentada por recuperación (RAG por sus siglas en inglés), abordando el desafío de evaluar tanto los componentes de recuperación como los de generación en estos sistemas. Proporciona métricas estructuradas como fidelidad, relevancia de la respuesta y utilización del contexto, que ayudan a evaluar la efectividad de los sistemas basados en RAG. Nuestros desarrolladores lo encontraron útil para realizar evaluaciones periódicas con el fin de afinar parámetros como las recuperaciones top-k y los modelos de incrustación. Algunos equipos han integrado Ragas en pipelines que se ejecutan diariamente, siempre que cambie la plantilla de prompts o el modelo. Aunque sus métricas ofrecen información valiosa, nos preocupa que el framework no capture todas las sutilezas e interacciones complejas de los pipelines RAG, y recomendamos considerar otros frameworks de evaluación adicionales. No obstante, Ragas destaca por su capacidad de optimizar la evaluación de RAG en entornos de producción, ofreciendo valiosas mejoras basadas en datos.

## 102. Score

### *Evaluar*

Muchas organizaciones que implementan sus propias plataformas internas de desarrollo tienden a crear sus propios sistemas de platform orchestration para hacer cumplir los estándares organizacionales entre los desarrolladores y sus equipos de alojamiento de plataformas. Sin embargo, las características básicas de una plataforma de despliegue pavimentada para alojar cargas de trabajo en contenedores de manera segura, consistente y conforme son similares de una organización a otra. ¿No sería genial si tuviéramos un lenguaje compartido para especificar esos requisitos? Score está mostrando cierto potencial para convertirse en un estándar en este ámbito. Es un lenguaje declarativo en formato YAML que describe cómo debe desplegarse una carga de trabajo en contenedores y qué servicios y parámetros específicos necesitará para ejecutarse. Score fue desarrollado originalmente por Humanitec como el lenguaje de configuración para su producto Platform Orchestrator pero ahora está bajo la custodia de la Cloud Native Computing Foundation (CNCF) como un proyecto de código abierto. Con el respaldo de la CNCF, Score tiene el potencial de ser más ampliamente utilizado más allá del producto de Humanitec. Ha sido lanzado con dos implementaciones de referencia: Kubernetes y Docker Compose. Esperamos que la extensibilidad de Score llevará a tener contribuciones de la comunidad para otras plataformas. Score ciertamente tiene un parecido con la especificación Open Application Model (OAM) para Kubevela, pero está más enfocado en el despliegue de cargas de trabajo en contenedores que en la aplicación completa. También hay cierta superposición con SST.dev, pero SST está más enfocado en el despliegue directo en una infraestructura en la nube en lugar de una plataforma interna de ingeniería. Estaremos observando con interés la evolución de Score.

## 103. shadcn

### *Evaluar*

shadcn desafía el concepto tradicional de bibliotecas de componentes al ofrecer componentes utilizables, de tipo copiar-y-pegar que se convierten en parte de tu código fuente. Este enfoque brinda a los equipos total propiedad y control, lo que permite una fácil personalización y extensión – áreas donde las librerías más convencionales y populares como MUI y Chakra UI a menudo quedan cortas. Creada con Radix UI y Tailwind CSS, shadcn se integra perfectamente en cualquier aplicación basada en React, lo que lo convierte en una buena opción para proyectos que priorizan el control y la extensibilidad. Incluye una CLI para ayudar en el proceso de copiar y pegar los componentes en el proyecto. Sus beneficios también incluyen la reducción de dependencias ocultas y evitar implementaciones estrechamente acopladas, razón por la cual shadcn está ganando terreno como una alternativa convincente para los equipos que buscan un enfoque más práctico y adaptable para el desarrollo front-end.

## 104. Slint

### *Evaluar*

Slint es un framework de GUI declarativo para construir interfaces de usuario nativas para aplicaciones en Rust, C++ o JavaScript. Aunque es un framework multiplataforma de UI con características importantes como vista previa en vivo, diseño de UI responsivo, integración con VS Code y una experiencia de usuario nativa, también nos gustaría resaltar principalmente su utilidad para sistemas embebidos. Los equipos que desarrollan aplicaciones embebidas han enfrentado tradicionalmente un número limitado de opciones para el desarrollo de la UI, cada una con sus propias implicaciones. Slint ofrece el balance perfecto entre la experiencia del desarrollador y el desempeño, utilizando un lenguaje de marcado fácil de usar, similar al HTML y compilado directamente a código máquina. En tiempo de ejecución, también cuenta con un bajo consumo de recursos, lo cual es crítico para los sistemas embebidos. En resumen, nos gusta Slint porque trae prácticas comprobadas del desarrollo web y móvil al ecosistema embebido.

## 105. SST

### *Evaluar*

SST es un framework para desplegar aplicaciones dentro del ambiente en la nube junto con proporcionar todos los servicios que la aplicación necesite para ejecutarse. SST no es solo una herramienta laC es un framework con una API en TypeScript que te permite definir el ambiente de tu aplicación, un servicio que despliega tu aplicación cuando se ejecuta un Git Push así como una consola GUI para administrar la aplicación resultante e invocar las funciones de administración de SST. Aunque SST estaba originalmente basado en AWS Cloud Formation y CDK, su última versión ha sido implementada sobre Terraform y Pulumi para que, en teoría, sea agnóstico a la nube. SST tiene soporte nativo para el despliegue de varios frameworks estandar de web, incluidos Next.js y Remix, pero también soporta aplicaciones API sin interfaz gráfica. SST parece pertenecer a una categoría propia. Si bien tiene cierta similitud con herramientas de orquestación de plataforma como Kubevela, también provee conveniencias para los desarrolladores, como un modo en vivo que envía las invocaciones de AWS Lambda a una función que se ejecuta en la máquina local del desarrollador. En este momento, SST sigue siendo una curiosidad, pero es un proyecto y una categoría de herramientas que vale la pena seguir a medida que evoluciona.

## 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



Thoughtworks es una consultora tecnológica global que integra estrategia, diseño e ingeniería para impulsar la innovación digital. Contamos con más de 10.500 empleados en 48 oficinas de 19 países. Durante los últimos 30 años, hemos logrado un impacto extraordinario junto con nuestros clientes, ayudándoles a resolver problemas empresariales complejos con la tecnología como elemento diferenciador.

 **thoughtworks**

Estrategia. Diseño. Ingeniería.