Technology Radar

Volume 30

Abril 2024

Um guia de opinião sobre o universo de tecnologia atual

/thoughtworks

Estratégia. Design. Engenharia.

Sobre o Radar

Thoughtworkers são pessoas apaixonadas por tecnologia. Nós desenvolvemos, pesquisamos, testamos, contribuímos com código livre, escrevemos sobre e visamos a sua constante melhoria — para todas as pessoas. Nossa missão é liderar e promover a excelência de software e revolucionar a indústria de TI. Nós criamos e compartilhamos o Technology Radar da Thoughtworks para apoiar essa missão. O Conselho Consultivo de Tecnologia (Technology Advisory Board, ou TAB), um grupo de líderes experientes em tecnologia da Thoughtworks, é responsável por criar o Radar. O grupo se reúne regularmente para discutir a estratégia global de tecnologia da empresa e as tendências tecnológicas que impactam significativamente a nossa indústria.

O Radar captura o resultado das discussões do TAB em um formato que procura oferecer valor a uma ampla gama de indivíduos interessados, de pessoas que desenvolvem software a CTOs. O conteúdo é concebido para ser um resumo conciso.

Nós encorajamos você a explorar essas tecnologias. O Radar é gráfico por natureza, agrupando os itens em técnicas, ferramentas, plataformas e linguagens & frameworks. No caso de itens que podem ser classificados em mais de um quadrante, escolhemos aquele que parece mais adequado. Além disso, agrupamos esses itens em quatro anéis para refletir nossas opiniões atuais em relação a cada um.

Para mais informações sobre o Radar, acesse: thoughtworks.com/pt/radar/faq.

Radar em um relance

A ideia por trás do Radar é rastrear coisas interessantes, que chamamos de blips. Organizamos blips no Radar usando duas categorias: quadrantes e anéis. Os quadrantes representam as diferentes naturezas dos blips. Os anéis indicam nossa recomendação para utilizar a tecnologia.

Um blip é uma tecnologia ou técnica que desempenha um papel no desenvolvimento de software. Os blips estão "em movimento" — suas posições no radar estão constantemente mudando — geralmente indicando que nossa confiança em recomendá-los tem crescido à medida que se movimentam entre os anéis.

Adote: Acreditamos firmemente que a indústria deveria adotar esses itens. Nós os usamos quando são apropriados em nossos projetos.

Experimente: Vale a pena ir atrás. É importante entender como desenvolver essa capacidade. As empresas devem testar esta tecnologia em um projeto que possa lidar com o risco.

Assess: Vale explorar com o objetivo de compreender como isso afetará sua empresa.

Evite: Prossiga com cautela.

Novo **Mudança de anel** **Sem alterações**

Nosso Radar é um olhar para o futuro. Para abrir o espaço para novos itens, apagamos itens que não foram modificados recentemente, o que não é um reflexo de seu valor, mas uma solução para nossa limitação de espaço.

Contribuições

O Conselho Consultivo de Tecnologia (em inglês, Technology Advisory Board - TAB) é um grupo formado por 22 tecnologistas experientes da Thoughtworks. O TAB se reúne duas vezes por ano pessoalmente e quinzenalmente por videochamada. Sua principal atribuição é ser um grupo consultivo para a nossa CTO Rachel Laycock, e a nossa CTO Emerita, Rebecca Parsons.

O TAB atua examinando tópicos que afetam soluções de tecnologia e tecnologistas da Thoughtworks. Esta edição do Thoughtworks Technology Radar é baseada em uma reunião do TAB realizada em Nova York, em Fevereiro de 2024.


Rebecca Parsons (CTO Emerita)


Rachel Laycock (CTO)


Martin Fowler (Chief Scientist)


Bharani Subramaniam


Birgitta Böckeler


Brandon Byars


Camilla Falconi Crispim


Erik Dörnenburg


Fausto de la Torre


Hao Xu


James Lewis


Marisa Hoenig


Maya Ormaza


Mike Mason

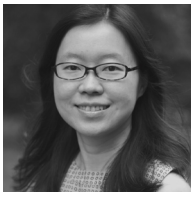

Neal Ford


Pawan Shah


Scott Shaw


Selvakumar Natesan


Shangqi Liu

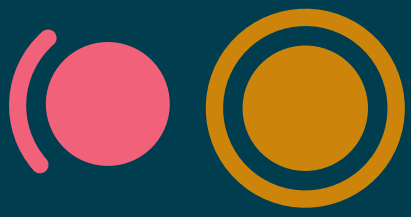

Sofia Tania


Vanya Seth


Will Amaral

Tradução: Alexandre Ramires, Bárbara Santos, Luiza Souza e Patrick Prado.

Temas



Licenças de código (quase) aberto

Dois tipos de discussões sobre licenças surgiram durante nossa reunião. O primeiro ponto diz respeito ao fato de que, por muitos anos, o ecossistema de desenvolvimento de software de código aberto confiou em um conjunto de licenças catalogadas pela Open Source Initiative (OSI), com um pequeno número de licenças populares usadas na maioria dos casos. Recentemente, no entanto, observamos uma agitação nessa paisagem anteriormente serena. Várias ferramentas de destaque receberam má publicidade recentemente, quando seus mantenedores mudaram — em alguns casos abruptamente — de uma licença de código aberto para um modelo comercial. Não temos nenhum problema em pagar por software e aceitamos o modelo comum de licenças comerciais para funcionalidades adicionais. No entanto, consideramos problemático quando a funcionalidade central de uma ferramenta amplamente utilizada é repentinamente colocada atrás de um paywall, especialmente quando um ecossistema se desenvolveu em torno da ferramenta. O segundo ponto de desenvolvimento interessante diz respeito ao software que se autoproclama código aberto, mas funcionalidades fundamentais só aparecem após o pagamento de assinaturas ou outras taxas por parte das pessoas usuárias. Embora esse modelo de negócio já existisse anteriormente, ele parece ser mais explorado com muitas das novas ferramentas de IA – oferecendo recursos incríveis, um pouco escondidos sob as letras miúdas. Aconselhamos especial diligência em relação às questões de licença. Preste atenção às ressalvas e certifique-se de que todos os arquivos em um repositório estejam cobertos pela licença no nível superior.

Times de desenvolvimento de software assistidos por IA

A Inteligência Artificial (IA) dominou nossas conversas, ocupando cerca de um terço dos tópicos abordados no Radar. Discutimos diversas ferramentas de IA voltadas para pessoas desenvolvedoras, como GitHub Copilot, CodiumAI, aider e Continue. Além disso, conversamos bastante sobre como o uso holístico de IA por toda a equipe modifica aspectos do desenvolvimento de software. Analisamos diversas ferramentas que não entraram na seleção final, incluindo terminais assistidos por IA como o Warp, a capacidade de converter screenshots em código, ChatOps baseados em LLMs e vários outros tópicos. Embora as ferramentas voltadas para desenvolvimento tendam a ter um grau maior de maturidade, suspeitamos que todos os aspectos do desenvolvimento de software possam se beneficiar gradualmente do uso pragmático de IA e ferramentas derivadas. Por isso, estamos acompanhando ativamente as inovações em todo o panorama do desenvolvimento. É claro que, com os novos recursos quase mágicos oferecidos pela IA, surgem também novos riscos para a qualidade e segurança do software. Isso exige que as equipes permaneçam vigilantes, incluindo manter pessoas não desenvolvedoras informadas sobre os potenciais perigos.

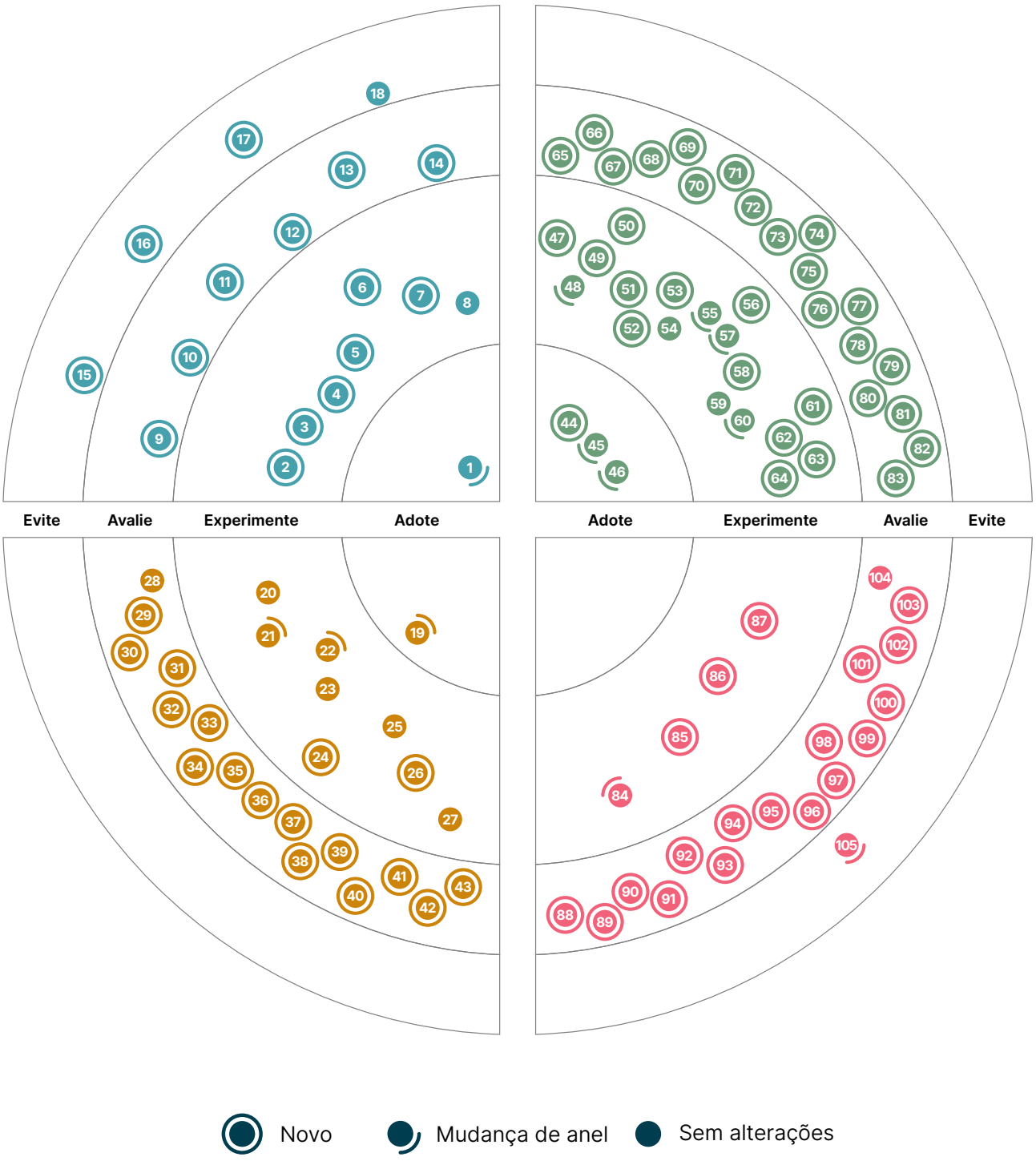
O surgimento de padrões de arquitetura para LLMs

Padrões são populares no mundo da tecnologia porque fornecem um nome sucinto para uma solução a um problema específico. Com o uso crescente dos modelos de linguagem de grande porte (LLMs), estamos começando a ver o surgimento de padrões de arquitetura específicos para suportar contextos comuns. Por exemplo, discutimos o NeMo Guardrails, que permite que as pessoas desenvolvedoras construam políticas de governança em torno do uso de LLM. Também falamos sobre ferramentas como Langfuse que permitem maior observabilidade nas etapas que levam à saída de um LLM e como lidar (e validar) bases de código infladas cheias de código gerado. Discutimos como a geração aumentada por recuperação (RAG) é nosso padrão preferido para melhorar a qualidade das saídas do LLM, especialmente no ecossistema corporativo. Além disso, discutimos técnicas como o uso de um LLM de menor potência (e custo) para produzir material que é seletivamente examinado por um LLM mais poderoso (e caro). Padrões formam um vocabulário importante para tecnologias, e esperamos ver uma explosão de padrões (e os inevitáveis antipadrões) à medida que a IA generativa continua a permear o desenvolvimento de software.

Aproximando pull requests de um CI adequado

A Thoughtworks sempre foi uma forte defensora de ciclos rápidos de feedback durante o desenvolvimento de software e, por isso, uma grande apoiadora da integração contínua (CI). Para auxiliar na adoção, construímos o primeiro servidor de CI da história — Cruise Control — que foi open-sourced no final dos anos 1990. Recentemente, nosso cientista-chefe, Martin Fowler, atualizou a definição canônica de integração contínua em seu bliki para renovar a atenção nessa prática.
No entanto, muitas de nossas equipes são obrigadas a ignorar a parte de CI do CI/CD porque se encontram em situações onde pull requests (PRs) são obrigatórias. Embora a prática de PRs tenha sido originalmente desenvolvida para gerenciar equipes de código aberto massivamente distribuídas e colaboradores não confiáveis, elas se tornaram um sinônimo de revisão por pares, mesmo em equipes de entrega pequenas e coesas. Nessas circunstâncias, muitos desenvolvedores anseiam pelo mesmo senso de fluxo que obtêm ao praticar a CI real. Analisamos diversas ferramentas que tentam aliviar as dores dos processos de revisão de PRs, incluindo gitStream e fila de merge do Github merge queue. Também discutimos técnicas como diferenciais empilhados que prometem se alinhar aos princípios centrais da CI, permitindo um controle mais granular sobre o processo de integração, e debatemos métodos para derivar métricas de PRs para identificar ineficiências e gargalos durante a entrega de software. O uso de ferramentas ajuda bastante nessa área devido à tendência de IA generativa para programação. Com assistentes de programação por IA, o rendimento de programação aumenta, o que leva à tendência de criar PRs maiores. Isso pressiona ainda mais os processos de revisão de código assíncronos. Mesmo que ainda prefiramos a prática original de CI, incentivamos as equipes que não podem usá-la devido a restrições externas a encontrar maneiras de melhorar a precisão da integração e a velocidade de seu ciclo de feedback.

O Radar



O Radar

Técnicas

Adote

1. Geração Aumentada por Recuperação (RAG)

Experimente

2. Geração automática de descritores de entidade Backstage
3. Combinando PLNs tradicionais com LLMs
4. Conformidade contínua
5. Funções de edge
6. Campeões em Segurança
7. De texto para SQL
8. Rastreando saúde em vez de dívida técnica

Avalie

9. Assistentes de equipe por IA
10. Análise de grafos para conversas baseadas em LLM
11. ChatOps baseados em LLM
12. Agentes autônomos impulsionados por LLM
13. Utilizando a IA Generativa para o entendimento de código legado
14. VISS

Evite

15. Testes de integração ampla
16. O uso excessivo de LLMs
17. Corrida para o fine-tuning de LLMs
18. Web components para aplicações web com SSR

Plataformas

Adote

19. CloudEvents

Experimente

20. Arm na nuvem
21. Aplicativos de Contêiner do Azure
22. Serviço de Open AI do Azure
23. DataHub
24. Plataformas de orquestração de infraestrutura
25. Pulumi
26. Rancher Desktop
27. Weights & Biases

Avalie

28. Bun
29. Chronosphere
30. DataOS
31. Dify
32. Elasticsearch Relevance Engine
33. FOCUS
34. Gemini Nano
35. HyperDX
36. IcePanel
37. Langfuse
38. Qdrant
39. RISC-V para embedded
40. Tigerbeetle
41. WebTransport
42. Zarf
43. ZITADEL

Evite

—

Ferramentas

Adote

44. Conan
45. Kaniko
46. Karpenter

Experimente

47. 42Crunch API Conformance Scan
48. actions-runner-controller
49. Contêiner do Emulador Android
50. AWS CUDOS
51. aws-nuke
52. Bruno
53. Develocity
54. GitHub Copilot
55. Gradio
56. Catálogo de versões do Gradle
57. Maestro
58. Ferramenta Microsoft SBOM
59. Open Policy Agent (OPA)
60. Runner auto-hospedado para GitHub Actions da Philips
61. Pop
62. Renovate
63. Terrascan
64. Velero

Avalie

65. aider
66. Akvorado
67. Baichuan 2
68. Cargo Lambda
69. Codium AI
70. Continue
71. Fern Docs
72. Granted
73. LinearB
74. LLaVA
75. Marimo
76. Mixtral
77. NeMo Guardrails
78. Ollama
79. OpenTofu
80. QAnything
81. System Initiative
82. Tetragon
83. Winglang

Evite

—

Linguagens e Frameworks

Adote

—

Experimente

84. Astro
85. DataComPy
86. Pinia
87. Ray

Avalie

88. Android Adaptability
89. Concrete ML
90. Crabviz
91. Crux
92. Databricks Asset Bundles
93. Electric
94. LiteLLM
95. LLaMA-Factory
96. MLX
97. Mojo
98. Otter
99. Pkl
100. Rust para o desenvolvimento de Interface para pessoas usuárias (UI)
101. vLLM
102. Voyager
103. WGPU
104. Zig

Evite

105. LangChain

Técnicas

Adote

1. Geração Aumentada por Recuperação (RAG)

Experimente

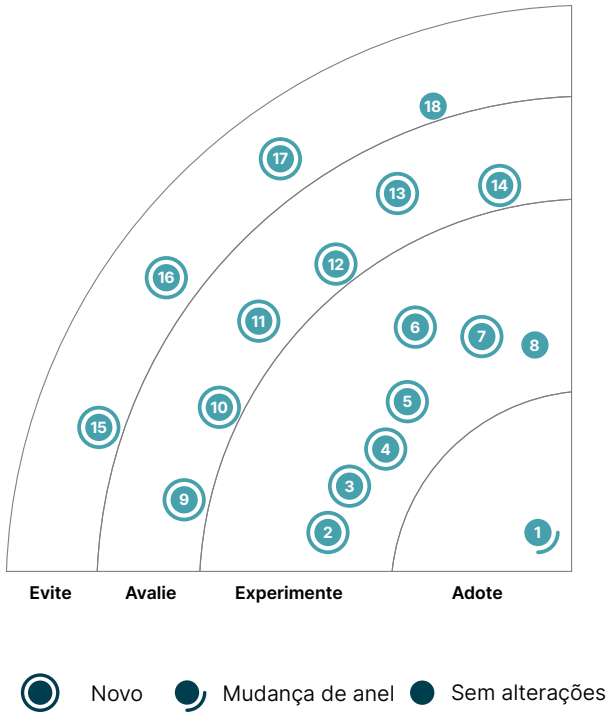
2. Geração automática de descritores de entidade Backstage
3. Combinando PLNs tradicionais com LLMs
4. Conformidade contínua
5. Funções de edge
6. Campeões em Segurança
7. De texto para SQL
8. Rastreando saúde em vez de dívida técnica

Avalie

9. Assistentes de equipe por IA
10. Análise de grafos para conversas baseadas em LLM
11. ChatOps baseados em LLM
12. Agentes autônomos impulsionados por LLM
13. Utilizando a IA Generativa para o entendimento de código legado
14. VISS

Evite

15. Testes de integração ampla
16. O uso excessivo de LLMs
17. Corrida para o fine-tuning de LLMs
18. Web components para aplicações web com SSR



1. Geração Aumentada por Recuperação (RAG)

Adote

A **geração aumentada por recuperação (RAG)** é o padrão preferido por nossas equipes para melhorar a qualidade das respostas geradas por um modelo de linguagem de grande porte (LLM). A técnica tem sido utilizada com sucesso em diversos projetos, incluindo a popular plataforma de IA Jugalbandi AI. Com a RAG, informações sobre documentos relevantes e confiáveis — em formatos como HTML e PDF — são armazenadas em bancos de dados que suportam um tipo de dados vetoriais ou pesquisa eficiente de documentos, como pgvector, Qdrant ou Elasticsearch Relevance Engine. Para um comando específico, o banco de dados é consultado para recuperar documentos relevantes, que são então combinados com o prompt para fornecer um contexto mais rico para o LLM. Isso resulta em saídas de alta qualidade e numa grande redução de alucinações (respostas irrelevantes). A janela de contexto — que determina o tamanho máximo da entrada do LLM — é limitada, o que significa que selecionar os documentos mais relevantes é crucial. Melhoramos a relevância do conteúdo que é adicionado ao comando por meio de uma nova classificação. Da mesma forma, os documentos geralmente são grandes demais para calcular um embedding, o que significa que eles devem ser divididos em partes menores. Esse costuma ser um problema complexo, e uma abordagem é fazer com que as partes se sobreponham até certo ponto.

2. Geração automática de descritores de entidade Backstage

Experimente

O **Backstage da Spotify** tem sido amplamente adotado por nossas clientes como a plataforma preferida para hospedar portais de experiência da pessoa desenvolvedora. O Backstage, por si só, é apenas um shell que hospeda plugins e fornece uma interface para gerenciar o catálogo de ativos que compõem o ecossistema de uma plataforma. Toda entidade a ser exibida ou gerenciada pelo Backstage é configurada no arquivo catalog-info, que contém dados como status, ciclo de vida, dependências e APIs, entre outros detalhes. Por padrão, os descritores individuais de entidade são escritos manualmente e geralmente mantidos e versionados pela equipe responsável pelo componente em questão. Manter os descritores atualizados pode ser tedioso e criar uma barreira para a adoção por parte das pessoas desenvolvedoras. Além disso, há sempre a possibilidade de alterações serem negligenciadas ou de alguns componentes serem completamente perdidos. Consideramos a **geração automática de descritores de entidade Backstage** uma forma mais eficiente e menos propensa a erros. A maioria das organizações possui fontes de informação existentes que podem impulsionar o processo de preenchimento de entradas do catálogo. Boas práticas de desenvolvimento — por exemplo, incluir tags apropriadas em recursos AWS ou adicionar metadados aos arquivos de código fonte — podem simplificar a descoberta de entidades e a geração de descritores. Esses processos automatizados podem então ser executados regularmente — uma vez ao dia, por exemplo — para manter o catálogo atualizado.

3. Combinando PLNs tradicionais com LLMs

Experimente

Os modelos de linguagem de grande porte (LLMs) são as facas suíças do processamento de linguagem natural (PLN). Porém, eles também são caros e nem sempre são a melhor ferramenta para o tarefa — às vezes, é mais eficaz usar uma tampa mais adequada. De fato, há muito potencial em **combinar PLN tradicionais com LLMs** , ou construir abordagens de PLN múltiplas em conjunto com LLMs para implementar casos de uso e alavancar suas capacidades para as etapas realmente necessárias. Abordagens tradicionais de ciência de dados e PLN para agrupamento de documentos, identificação e classificação de tópicos e até mesmo sumarização são mais baratas e podem ser mais eficazes para resolver parte do problema do seu caso de uso. Em seguida, usamos LLMs quando precisamos gerar e resumir textos longos, ou combinar vários documentos grandes, para aproveitar a atenção e memória superiores do LLM. Por exemplo, usamos com sucesso essa combinação de técnicas para gerar um relatório de tendências abrangente para um domínio a partir de um grande corpus de documentos de tendências individuais, utilizando agrupamento tradicional juntamente com o poder de geração de LLMs.

4. Conformidade contínua

Experimente

A **conformidade contínua** é a prática de garantir, de forma contínua e com alto grau de automatização, que os processos e tecnologias de desenvolvimento de software estejam em conformidade com regulamentações da indústria e padrões de segurança. Verificar manualmente as vulnerabilidades de segurança e aderir a regulamentações pode retardar o desenvolvimento e introduzir erros. Como alternativa, as organizações podem automatizar verificações e auditorias de conformidade. Elas podem integrar ferramentas aos pipelines de desenvolvimento de software, permitindo às equipes detectar e abordar problemas de conformidade desde o início do processo. A codificação de regras de conformidade e melhores práticas auxilia na aplicação consistente de políticas e padrões em todas as equipes. Isso permite a verificação de alterações de código em busca de vulnerabilidades, a aplicação de padrões de codificação e o rastreamento de alterações na configuração da infraestrutura para garantir que atendam aos requisitos de compliance. Por fim, a geração automatizada de relatórios dos itens acima simplifica as auditorias e fornece evidências claras de conformidade. Já discutimos técnicas como a publicação de SBOMs e a aplicação das recomendações do SLSA — elas podem ser excelentes pontos de partida. Os benefícios dessa técnica são múltiplos. Primeiramente, a automação leva a softwares mais seguros ao identificar e mitigar vulnerabilidades precocemente. Em segundo lugar, os ciclos de desenvolvimento se aceleram com a eliminação de tarefas manuais. Custos reduzidos e consistência aprimorada são vantagens adicionais. Para as indústrias críticas à segurança, como a de veículos guiados por software, a conformidade contínua automatizada pode melhorar a eficiência e a confiabilidade do processo de certificação, resultando em veículos mais seguros e confiáveis nas estradas.

5. Funções de edge

Experimente

Embora não seja um conceito novo, temos observado a crescente disponibilidade e adoção da execução descentralizada de código por meio de redes de distribuição de conteúdo (CDNs). Serviços como Funções de Edge do Cloudflare Workers ou do Amazon CloudFront Edge Functions fornecem um mecanismo para executar trechos de código sem servidor próximos à localização geográfica da pessoa usuária. As **funções de edge** não apenas oferecem menor latência se uma resposta puder ser gerada na borda da rede, mas também apresentam a oportunidade de reescrever requisições e respostas de maneira específica para cada localidade, à medida que elas trafegam de e para o servidor regional. Por exemplo, você pode reescrever a URL de uma requisição para rotear para um servidor específico que possui dados locais relevantes a um campo encontrado no corpo da requisição. Essa abordagem é mais adequada para processos sem estado, curtos e de execução rápida, pois a capacidade computacional na borda é limitada.

6. Campeões em Segurança

Experimente

Os **Campeões em Segurança** são membros da equipe que pensam criticamente sobre as repercussões de segurança em decisões de entrega técnicas e não técnicas. Eles levantam essas questões e preocupações com a liderança da equipe e possuem um sólido entendimento das diretrizes e requisitos básicos de segurança. Eles ajudam as equipes de desenvolvimento a abordar todas as atividades durante a entrega de software com uma mentalidade de segurança, reduzindo assim os riscos gerais de segurança para os sistemas que desenvolvem. Um campeão de segurança não é uma posição separada, mas uma responsabilidade atribuída a um membro existente da equipe, orientado por treinamento apropriado de profissionais de segurança. Equipados com esse treinamento, os campeões de segurança melhoram a consciência de segurança da equipe disseminando conhecimento e atuando como uma ponte entre as equipes de desenvolvimento e segurança. Um ótimo exemplo de atividade que os campeões de segurança podem ajudar a impulsionar dentro da equipe é a modelagem de ameaças, que auxilia as equipes a pensar nos riscos de segurança desde o início do projeto. Indicar e treinar um campeão de segurança na equipe é um ótimo primeiro passo, mas confiar apenas nos campeões sem o comprometimento adequado dos líderes pode levar a problemas. Construir uma mentalidade de segurança, em nossa experiência, requer comprometimento de toda a equipe e gerentes.

7. De texto para SQL

Experimente

De texto para SQL é uma técnica que converte as consultas feitas em linguagem natural para consultas e SQL executáveis por um banco de dados. Embora os modelos de linguagem de grande porte (LLMs) consigam entender e transformar linguagem natural, criar um SQL preciso para o seu próprio esquema pode ser desafiador. É neste ponto que entra o Vanna, um framework de código aberto em geração aumentada por recuperação (RAG) em Python para geração de SQL. O Vanna funciona em duas etapas: primeiro você cria embeddings com as instruções da linguagem de definição de dados (DDLs) e amostras de SQL para o seu esquema, e depois formula perguntas em linguagem natural. Ainda que o o Vanna funcione com qualquer LLM, incentivamos a avaliação do NSQL, um LLM específico de domínio para tarefas de texto para SQL.

8. Rastreando saúde em vez de dívida técnica

Experimente

Estamos observando que as equipes estão melhorando seu ecossistema ao tratar a classificação de saúde da mesma forma que outros objetivos de nível de serviço (SLO) e priorizando as melhorias de acordo, em vez de se concentrarem apenas no rastreamento da dívida técnica. Ao alocar recursos de forma eficiente para resolver os problemas de maior impacto relacionados à saúde, as equipes e organizações podem reduzir os custos de manutenção de longo prazo e evoluir os produtos de forma mais eficiente. Essa abordagem também aprimora a comunicação entre stakeholders técnicos e não técnicos, promovendo uma compreensão comum do estado do sistema. Embora as métricas possam variar entre as organizações (veja este post do blog para exemplos), elas contribuem para a sustentabilidade de longo prazo e garantem que o software permaneça adaptável e competitivo. Em um cenário digital em constante mudança, **rastrear a saúde em vez da dívida técnica** dos sistemas fornece uma estratégia estruturada e baseada em evidências para mantê-los e melhorá-los.

9. Assistentes de equipe por IA

Avalie

Ferramentas assistentes de programação por IA, como o GitHub Copilot, atualmente são discutidas principalmente no contexto de auxiliar e aprimorar o trabalho individual. No entanto, a entrega de software é, e continuará sendo um trabalho de equipe. Por isso, é importante buscar maneiras de criar **assistentes de equipe por IA** para ajudar a construir o time 10x, ao invés de grupos de engenheiros 10x isolados com assistência de IA . Começamos a utilizar uma abordagem de assistência de equipe que pode aumentar a amplificação do conhecimento, a qualificação e o alinhamento por meio de uma combinação de comando e fontes de conhecimento. Os comandos padronizados facilitam o uso de práticas recomendadas acordadas no contexto da equipe, como técnicas e modelos para escrita de histórias de usuários ou a implementação de práticas como modelagem de ameaças. Além dos comandos, as fontes de conhecimento disponibilizadas por meio de geração aumentada por recuperação fornecem informações contextualmente relevantes de diretrizes organizacionais ou bases de conhecimento específicas do setor. Essa abordagem dá aos membros da equipe acesso ao conhecimento e aos recursos de que precisam no momento exato.

10. Análise de grafos para conversas baseadas em LLM

Avalie

Chatbots baseados em modelos de linguagem de grande porte (LLMs) estão ganhando muita popularidade e estamos observando técnicas emergentes para colocá-los em produção. Um dos desafios da produtização é entender como as pessoas estão conversando com um chatbot orientado por algo tão genérico como um LLM, onde a conversa pode seguir por muitos caminhos. Compreender a realidade dos fluxos de conversação é crucial para melhorar o produto e as taxas de conversão. Uma técnica para enfrentar esse problema é a **análise de grafos para conversas baseadas em LLM.** Os agentes que suportam um chat com um objetivo específico - como uma ação de compra ou a resolução bem-sucedida do problema de uma cliente - geralmente podem ser representados como uma máquina de estados desejada. Ao carregar todas as conversas em um grafo, você pode analisar os padrões reais e procurar discrepâncias em relação à máquina de estados esperada. Isso ajuda a encontrar bugs e oportunidades para melhoria do produto.

11. ChatOps baseados em LLM

Avalie

ChatOps baseados em LLM é uma aplicação emergente desses modelos por meio de plataformas de chat (principalmente o Slack) que permite a pessoas engenheiras construir, implantar e operar software através de linguagem natural. Isso tem o potencial de simplificar os fluxos de trabalho de engenharia, aprimorando a descoberta e a usabilidade dos serviços de plataforma. Os dois exemplos iniciais na época de redação deste blip são o PromptOps e o Kubiya. No entanto, considerando a precisão necessária para ambientes de produção, as organizações devem avaliar minuciosamente essas ferramentas antes de permitir a inclusão em seu processo de produção.

12. Agentes autônomos impulsionados por LLM

Avalie

Agentes autônomos impulsionados por LLM estão evoluindo além de agentes isolados e sistemas multiagentes estáticos com o surgimento de frameworks como o Autogen e o CrewAI. Essas estruturas permitem que as pessoas usuárias definam agentes com funções específicas, atribuam tarefas e habilitem a colaboração entre agentes para concluir essas tarefas por meio de delegação ou conversação. Semelhante aos sistemas de agente único que surgiram anteriormente, como o AutoGPT, agentes individuais podem dividir tarefas, utilizar ferramentas pré-configuradas e solicitar intervenção humana. Embora ainda esteja nos estágios iniciais de desenvolvimento, esta área está se desenvolvendo rapidamente e tem um potencial empolgante para exploração.

13. Utilizando a IA Generativa para o entendimento de código legado

Avalie

A IA Generativa (GenAI) e os modelos de linguagem de grande porte (LLMs) podem auxiliar pessoas desenvolvedoras tanto a escrever quanto a entender código. Na prática, a aplicação atual se limita principalmente a trechos de código menores, mas novos produtos e avanços tecnológicos estão surgindo para **utilizar a IA Generativa no entendimento de código legado**. Isso é particularmente útil para bases de código antigas mal documentadas ou cuja documentação esteja desatualizada ou imprecisa. Por exemplo, o Driver AI or bloop usam RAGs que combinam inteligência de linguagem e busca de código com LLMs para auxiliar as pessoas usuárias a navegarem em uma base de código. Modelos emergentes com janelas de contexto cada vez maiores também ajudarão a tornar essas técnicas mais viáveis para bases de código de grande porte. Outra aplicação promissora da IA Generativa para código legado está na modernização de mainframes, onde gargalos frequentemente se formam em torno de pessoas engenheiras reversas que precisam entender a base de código existente e transformar esse entendimento em requisitos para o projeto de modernização. O uso da IA Generativa para auxiliar essas pessoas engenheiras reversas pode acelerar a conclusão do seu trabalho.

14. VISS

Avalie

A Zoom recentemente tornou seu Sistema de Pontuação de Impacto de Vulnerabilidade, ou **VISS,** em um sistema de código aberto. Esse sistema foca principalmente na pontuação de vulnerabilidades que prioriza medidas de segurança comprovadamente efetivas. O VISS difere do Sistema Comum de Pontuação de Vulnerabilidade (CVSS) ao não se concentrar em cenários de pior caso e tentar mensurar de forma mais objetiva o impacto das vulnerabilidades da perspectiva do defensor. Para isso, o VISS fornece uma interface web (UI) para calcular a pontuação da vulnerabilidade com base em diversos parâmetros — categorizados em grupos de plataforma, infraestrutura e dados — incluindo o impacto na plataforma, o número de tenants afetados, impacto nos dados e muito mais. Embora ainda não tenhamos muita experiência prática com esta ferramenta específica, acreditamos que esse tipo de abordagem de avaliação prioritária baseada no setor e no contexto vale a pena ser praticada.

15. Testes de integração ampla

Evite

Apesar de aplaudirmos o foco em testes automatizados, seguimos observando diversas organizações investindo excessivamente no que consideramos **testes de integração ampla** ineficazes. Como o termo teste de integração é ambíguo, adotamos a classificação ampla do verbete de Martin Fowler sobre o assunto em seu bliki. Ele indica que esse tipo de teste requer versões ativas de todas as dependências de tempo de execução. Obviamente, esse tipo de teste é caro, pois exige um ambiente de teste completo com toda a infraestrutura, dados e serviços necessários. Gerenciar as versões corretas de todas essas dependências demanda uma sobrecarga significativa de coordenação, o que tende a retardar os ciclos de lançamento. Além disso, os próprios testes geralmente são frágeis e pouco úteis. Por exemplo, determinar se a falha de um teste ocorreu devido ao código novo, incompatibilidade de versões de dependências ou ao ambiente pode exigir bastante esforço, e a mensagem de erro raramente ajuda a identificar a origem do problema. Essas críticas não significam que discordamos de testes de integração automatizados do tipo caixa preta em geral. No entanto, consideramos uma abordagem mais vantajosa aquela que equilibra a necessidade de confiança com a frequência de lançamento. Isso pode ser feito em duas etapas, validando o comportamento do sistema sob teste. Primeiro, validamos o comportamento do sistema considerando um determinado conjunto de respostas das dependências de tempo de execução. Utilizamos virtualização de serviços para criar dublês de teste para essas dependências, o que simplifica o gerenciamento de dados de teste e permite testes determinísticos. Em seguida, validamos essas premissas de ambiente com dependências reais usando testes de contrato.

16. O uso excessivo de LLMs

Evite

Na pressa para aproveitar o que há de mais recente em IA, muitas organizações estão adotando rapidamente modelos de linguagem de grande porte (LLMs) para diversas aplicações, desde geração de conteúdo até processos complexos de tomada de decisão. O fascínio pelos LLMs é inegável; eles oferecem uma solução aparentemente sem esforço para problemas complexos, e as pessoas desenvolvedoras muitas vezes podem criar tal solução rapidamente e sem a necessidade de anos de experiência em aprendizado de máquina (ML) profundo. Pode ser tentador lançar uma solução baseada em LLM assim que ela esteja mais ou menos funcional e seguir em frente. Embora essas provas de conceito baseadas em LLM sejam úteis, aconselhamos as equipes a analisarem cuidadosamente para que a tecnologia está sendo usada e a considerarem se um LLM é realmente a solução final correta. Muitos problemas que um LLM pode resolver — como análise de sentimento ou classificação de conteúdo — podem ser resolvidos de forma mais barata e fácil usando o Processamento de Linguagem Natural (PLN) tradicional. Analisar o que o LLM está fazendo e, em seguida, analisar outras soluções potenciais não apenas mitiga os riscos associados ao **uso excessivo de LLMs**, mas também promove uma compreensão e aplicação mais matizadas das tecnologias de IA.

17. Corrida para o fine-tuning de LLMs

Evite

À medida que organizações buscam formas de fazer com que os modelos de linguagem de grande porte (LLMs) funcionem no contexto de seus produtos, domínios ou conhecimento organizacional, estamos vendo uma **corrida para o fine-tuning de LLMs.** Embora o fine-tuning possa ser uma ferramenta poderosa para aumentar a especificidade de tarefas em um caso de uso, em muitos casos ele não é necessário. Um dos erros mais comuns nessa pressa pelo fine-tuning é tentar tornar um aplicativo baseado em LLM ciente de conhecimento e fatos específicos ou do código-base de uma organização. Na grande maioria desses casos, usar uma forma de geração aumentada por recuperação (RAG) oferece uma solução melhor e uma relação custo-benefício mais vantajosa. O fine-tuning requer recursos computacionais consideráveis e expertise, além de introduzir desafios ainda maiores relacionados a dados sensíveis e proprietários do que a RAG. Há também o risco de subajuste (underfitting), quando não há dados suficientes para o fine-tuning, ou, menos frequentemente, de superajuste (overfitting), quando há dados em excesso, o que resulta em um desbalanceamento na especificidade de tarefas que você precisa. Analise atentamente esses prós e contras e considere as alternativas antes de se apressar para fazer o fine-tuning de um LLM para o seu caso de uso.

18. Web components para aplicações web com SSR

Evite

Com a adoção de frameworks como Next.js e htmx, observamos um aumento no uso de renderização no lado do servidor (SSR). Como tecnologia do navegador, utilizar web components não é trivial. Frameworks surgiram para facilitar isso, alguns até mesmo usando um motor de navegador, mas a complexidade ainda persiste. Nossas equipes de desenvolvimento se vêem precisando de soluções alternativas e esforço adicional para ordenar componentes de front-end e componentes do lado do servidor. Pior do que a experiência da pessoa desenvolvedora é a experiência da pessoa usuária: o desempenho do carregamento da página é prejudicado quando web components customizados precisam ser carregados e hidratados no navegador, e mesmo com pré-renderização e ajustes cuidadosos do componente, um lampejo de conteúdo sem estilo ou alguma mudança de layout é quase inevitável. Conforme mencionado no Radar anterior, uma de nossas equipes precisou migrar seu design system do Stencil, baseado em web components, devido a esses problemas. Recentemente, recebemos relatos de outra equipe que acabou substituindo componentes gerados no lado do servidor por componentes do lado do navegador devido à complexidade de desenvolvimento. Alertamos contra o uso de **web components para aplicações web com SSR** , mesmo que sejam suportados por frameworks.

Plataformas

Adote

19. CloudEvents

Experimente

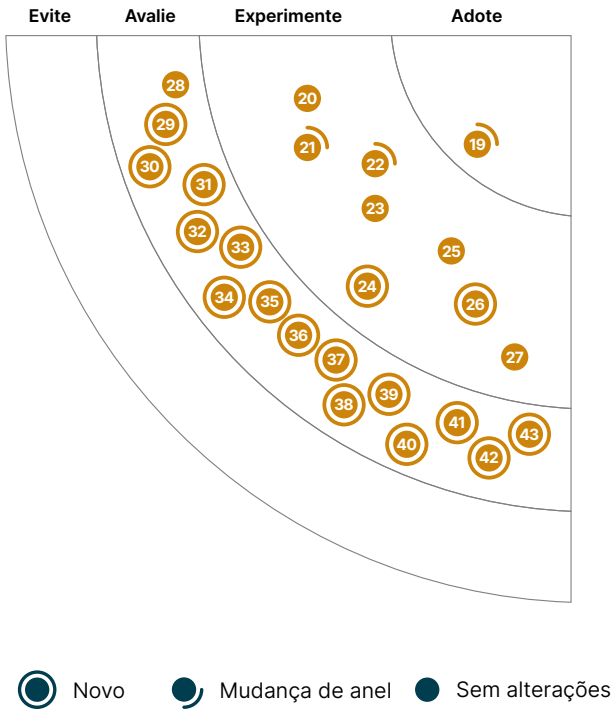
20. Arm na nuvem
21. Aplicativos de Contêiner do Azure
22. Serviço de Open AI do Azure
23. DataHub
24. Plataformas de orquestração de infraestrutura
25. Pulumi
26. Rancher Desktop
27. Weights & Biases

Avalie

28. Bun
29. Chronosphere
30. DataOS
31. Dify
32. Elasticsearch Relevance Engine
33. FOCUS
34. Gemini Nano
35. HyperDX
36. IcePanel
37. Langfuse
38. Qdrant
39. RISC-V para embedded
40. Tigerbeetle
41. WebTransport
42. Zarf
43. ZITADEL

Evite

—



19. CloudEvents

Adote

Eventos são mecanismos comuns em arquiteturas baseadas em eventos ou em aplicações sem servidor. Entretanto, produtores e provedores de nuvem tendem a oferecer suporte a eles em diferentes formatos, o que impede a interoperabilidade entre plataformas e infraestruturas. O **CloudEvents** é uma especificação para descrever dados de eventos em formatos comuns para fornecer interoperabilidade entre serviços, plataformas e sistemas. Ele fornece SDKs em várias linguagens para que você possa incorporar a especificação em seu aplicativo ou chain de ferramentas. Nossas equipes o utilizam não apenas para fins de plataformas em nuvem cruzadas, mas também para especificação de eventos de domínio, entre outros cenários. O CloudEvents é hospedado pela Cloud Native Computing Foundation (CNCF) e agora é um projeto graduado. Nossas equipes adotam o CloudEvents como padrão para construir arquiteturas baseadas em eventos e por esse motivo estamos movendo-o para Adoção.

20. Arm na nuvem

Experimente

As instâncias de computação Arm na nuvem se tornaram cada vez mais populares nos últimos anos devido ao seu custo e eficiência energética em comparação com as instâncias tradicionais baseadas em x86. Muitos provedores de nuvem agora oferecem instâncias baseadas em Arm, incluindo AWS, Azure e GCP. Os benefícios de custo ao executar Arm na nuvem podem ser particularmente vantajosos para empresas que executam grandes cargas de trabalho ou precisam escalar. Estamos observando muitas equipes migrando para instâncias Arm para cargas de trabalho como serviços JVM e até mesmo bancos de dados (incluindo RDS) sem nenhuma alteração no código e com modificações mínimas nos scripts de compilação. Novos aplicativos e sistemas baseados em nuvem estão cada vez mais optando por Arm como padrão. Com base em nossas experiências, recomendamos instâncias de computação Arm para todas as cargas de trabalho, a menos que haja dependências específicas da arquitetura. As ferramentas para suportar arquiteturas múltiplas, como imagens Docker multi-arquitetura, também simplificam os fluxos de trabalho de compilação e implantação.

21. Aplicativos de Contêiner do Azure

Experimente

O **Aplicativos de Contêiner do Azure** é uma plataforma de aplicativo gerenciada baseada em Kubernetes que simplifica a implantação de cargas de trabalho em contêineres. Comparado ao Azure Kubernetes Service (AKS), o Aplicativos de Contêiner do Azure reduz a sobrecarga operacional e administrativa da execução de aplicativos em contêineres. No entanto, isso ocorre em detrimento de alguma flexibilidade e controle, o que é um trade-off que as equipes precisam considerar. Outro produto nessa área, o Azure Container Instances, geralmente é muito limitado para uso em produção. Nossas equipes começaram a utilizar o Aplicativos de Contêiner do Azure no ano passado, quando ainda estava em preview pública, obtendo bons resultados, mesmo ao executar contêineres grandes. Agora que está disponível para o público em geral, estamos considerando-o para mais casos de uso. Ambos os recursos, Dapr e o KEDA Autoscaler, são suportados.

22. Serviço de Open AI do Azure

Experimente

O **Serviço de Open AI do Azure** fornece acesso ao GPT-4, GPT-3.5-Turbo, Embeddings, modelo DALL-E e outros da OpenAI por meio de uma API REST, um SDK Python e uma interface baseada na web. Os modelos podem ser adaptados para tarefas como geração de conteúdo, sumarização, busca semântica e tradução de linguagem natural para código. O ajuste fino também está disponível por meio de aprendizado de poucos exemplos e personalização de hiperparâmetros. Em comparação com a API própria da OpenAI, o Serviço OpenAI do Azure se beneficia dos recursos de segurança e conformidade de nível corporativo do Azure, está disponível para mais regiões (embora a disponibilidade seja limitada para cada uma das grandes regiões geográficas) e suporta rede privada, filtragem de conteúdo e controle de versão manual do modelo. Por esses motivos e por nossa experiência positiva com ele, recomendamos que as empresas que já utilizam o Azure considerem usar o Serviço OpenAI do Azure em vez da API da OpenAI.

23. DataHub

Experimente

Ao construir produtos de dados utilizando o conceito de mentalidade para produtos de dados, é essencial considerar a linhagem da dados, sua descoberta e governança. Nossos times descobriram que o **DataHub** oferece suporte particularmente útil nesses quesitos. Embora versões anteriores do DataHub exigissem a bifurcação (fork) e o gerenciamento da sincronização a partir do produto principal (caso fosse necessária a atualização do modelo de metadados), melhorias em lançamentos recentes trouxeram recursos que permitem que nossos times implementem modelos de metadados customizados com uma arquitetura baseada em plugins. Outra funcionalidade útil do DataHub é a robusta linhagem de dados ponta-a-ponta, da origem ao processamento e consumo. O DataHub suporta integração baseada em push e também extração de linhagem baseada em pull, que automaticamente varre os metadados técnicos em fontes de dados, agendadores, orquestradores (como Airflow DAG scanning), tarefas de pipelines de processamento e painéis, entre outros. Como uma opção de código aberto para um catálogo de dados holístico, o DataHub está se tornando a escolha padrão para nossos times.

24. Plataformas de orquestração de infraestrutura

Experimente

O código-fonte interno para orquestração de infraestrutura frequentemente se torna um desperdício de tempo para manutenção e resolução de problemas. É nesse contexto que estão surgindo **plataformas de orquestração de infraestrutura,** prometendo padronizar e transformar em produtos os diversos aspectos do ciclo de vida de entrega e implantação de código de infraestrutura. Isso inclui ferramentas de build como Terragrunt e Terraspace, serviços de fornecedores de ferramentas IaC como Terraform Cloud e Pulumi Cloud, e também plataformas e serviços agnósticos a ferramentas como env0 e Spacelift. Existe um rico ecossistema de ferramentas e serviços de orquestração específicos para Terraform, frequentemente chamados de TACOS (Terraform Automation and Collaboration Software), incluindo Atlantis, Digger, Scalr, Terramate e Terrateam. Cada uma dessas plataformas possibilita diferentes fluxos de trabalho, incluindo GitOps, Continuous Delivery e compliance as code (conformidade como código). Estamos otimistas com o crescimento de soluções nesta área. Recomendamos que as equipes de engenharia de infraestrutura e plataforma explorem como utilizar essas plataformas para reduzir a quantidade de código customizado não-diferencial que precisam desenvolver e manter para sua infraestrutura. A padronização de como o código de infraestrutura é estruturado, compartilhado, entregue e implantado também deve criar oportunidades para o surgimento de um ecossistema de ferramentas compatíveis para testes, medição e monitoramento de infraestrutura.

25. Pulumi

Experimente

O ecossistema de ferramentas para o gerenciamento de infraestrutura como código segue em constante evolução, e ficamos contentes em ver que o **Pulumi** não é exceção. A plataforma recentemente adicionou suporte para Java e YAML, para gerenciamento de infraestrutura em escala, assim como para uma variedade de configurações e integrações com a nuvem, tornando a plataforma ainda mais atraente. Para nossas equipes, o Pulumi continua sendo a principal alternativa ao Terraform para o desenvolvimento de código para múltiplas plataformas de nuvem.

26. Rancher Desktop

Experimente

As mudanças no licenciamento do Docker Desktop nos levaram a buscar alternativas para executar uma frota de contêineres na máquina local das pessoas desenvolvedoras. Recentemente, tivemos um bom resultado com o **Rancher Desktop**. Este aplicativo gratuito e de código aberto é relativamente fácil de baixar e instalar em máquinas Apple, Windows ou Linux e oferece um prático cluster local do Kubernetes com uma interface gráfica (GUI) para configuração e monitoramento. Embora o Colima tenha se tornado nossa alternativa preferida ao Docker Desktop, ele é principalmente uma ferramenta de linha de comando (CLI). Por outro lado, o Rancher Desktop atrairá aquelas pessoas que não querem abrir mão da interface gráfica oferecida pelo Docker Desktop. Assim como o Colima, o Rancher Desktop permite escolher entre dockerd ou containerd como o runtime de contêiner subjacente. A opção de usar o containerd diretamente libera a dependência do Docker CLI, mas a opção dockerd oferece compatibilidade com outras ferramentas que dependem dele para se comunicar com o daemon de execução.

27. Weights & Biases

Experimente

Weights & Biases é uma plataforma de aprendizado de máquina (ML) que auxilia no desenvolvimento mais ágil de modelos por meio do rastreamento de experimentos, controle de versão de conjuntos de dados, visualização do desempenho do modelo e gerenciamento de modelos. A plataforma pode ser integrada ao código de ML existente para acesso das métricas em tempo real, logs de terminal e estatísticas do sistema transmitidas ao painel para análises posteriores. Recentemente, o Weights & Biases expandiu sua atuação para a observabilidade de LLM com o Traces. O Traces visualiza o fluxo de execução de cadeias de prompts, assim como entradas e saídas intermediárias, além de fornecer metadados sobre a execução da cadeia (como tokens usados e horário de início e término). Nossas equipes consideram a ferramenta útil para depuração e obtenção de uma melhor compreensão da arquitetura da cadeia.

28. Bun

Avalie

Bun é um novo runtime JavaScript, similar ao Node.js ou Deno. Ao contrário do Node.js ou Deno, o Bun é construído usando o JavaScriptCore do WebKit em vez do motor V8 do Chrome. Projetado como uma substituição direta para o Node.js, o Bun é um único binário (escrito em Zig) que funciona como bundler, transpilador e gerenciador de pacotes para aplicações JavaScript e TypeScript. Desde o nosso último volume, o Bun passou da fase beta para uma versão estável 1.0. O Bun foi construído do zero com diversas otimizações — incluindo inicialização rápida, renderização do lado do servidor aprimorada e um gerenciador de pacotes alternativo muito mais rápido — e por isso, incentivamos que você o avalie como seu motor de execução JavaScript.

29. Chronosphere

Avalie

Ao gerenciar arquiteturas distribuídas, contabilizar o custo de ordenação, indexação e acesso a dados é tão crítico quanto a observabilidade. O **Chronosphere** adota uma abordagem única para o gerenciamento de custos, rastreando o uso de dados de observabilidade para que as organizações possam considerar as compensações entre custo e valor de várias métricas. Com a ajuda do Analisador de Uso de Métricas, que faz parte do Plano de Controle do Chronosphere, as equipes podem identificar e excluir métricas que raramente (ou nunca) usam, gerando assim economias significativas de custo ao reduzir a quantidade de dados que as organizações precisam vasculhar. Dadas essas vantagens, somadas à capacidade do Chronosphere de corresponder à funcionalidade de outras ferramentas de observabilidade para soluções hospedadas em nuvem, consideramos que seja uma opção atraente para organizações analisarem.

30. DataOS

Avalie

Com a crescente adoção do data mesh, nossas equipes estão em busca de plataformas de dados que tratem os produtos de dados como entidades de primeira classe. O **DataOS** é um desses produtos. Ele oferece gerenciamento de ciclo de vida completo para projetar, construir, implantar e evoluir produtos de dados. O DataOS fornece especificações declarativas padronizadas escritas em YAML, que abstraem a complexidade de baixo nível da configuração de infraestrutura e permitem que as pessoas desenvolvedoras definam os produtos de dados facilmente via CLI/API. Suporte a políticas de controle de acesso com ABAC e políticas de dados para filtrar e mascarar dados. Capacidade de federar dados em uma variedade de fontes de dados, o que reduz a duplicação de dados e a movimentação para um local central. O DataOS se encaixa melhor em cenários greenfield, pois fornece uma solução pronta para uso para governança de dados, descoberta de dados, gerenciamento de recursos de infraestrutura e observabilidade. Para cenários brownfield, a capacidade de orquestrar recursos fora do DataOS (por exemplo, data stacks como Databricks) ainda está em fase inicial de evolução. Se o seu ecossistema não tem uma forte opinião sobre ferramentas de dados, o DataOS é uma boa maneira de agilizar o processo de construção, implantação e consumo de produtos de dados de forma completa.

31. Dify

Avalie

Dify é uma plataforma com interface gráfica para desenvolvimento de aplicativos com modelos de linguagem de grande porte (LLMs), tornando a prototipação ainda mais acessível. Ele suporta o desenvolvimento de apps de chat e geração de texto com modelos de comandos. Além disso, o Dify suporta geração aumentada por recuperação com conjuntos de dados importados e pode funcionar com vários modelos. Estamos entusiasmados com essa categoria de aplicativos. No entanto, com base em nossa experiência, o Dify ainda não está totalmente pronto para uso, pois alguns recursos apresentam bugs ou não parecem totalmente finalizados. No momento, porém, não conhecemos nenhum concorrente que seja melhor.

32. Elasticsearch Relevance Engine

Avalie

Apesar do crescente interesse em vector databases para casos de uso de geração aumentada por recuperação (RAG), pesquisas e relatórios de experiência sugerem que a combinação de busca tradicional de texto completo com busca vetorial (em uma busca híbrida) pode gerar resultados superiores. Por meio do **Elasticsearch Relevance Engine (ESRE)**, a consagrada plataforma de busca de texto completo Elasticsearch suporta modelos de embedding incorporados e personalizados, busca vetorial e busca híbrida com mecanismos de classificação como o Reciprocal Rank Fusion. Embora essa área ainda esteja amadurecendo, em nossa experiência, o uso desses recursos do ESRE juntamente com os recursos tradicionais de filtragem, ordenação e classificação fornecidos pelo Elasticsearch tem produzido resultados promissores, sugerindo que plataformas de busca estabelecidas que suportam busca semântica não devem ser negligenciadas.

33. FOCUS

Avalie

Dados de faturamento de nuvem e SaaS podem ser complexos, inconsistentes entre fornecedores e difíceis de entender. A Especificação de Custo e Uso Aberto FinOps (**FOCUS**) visa reduzir esse atrito com uma especificação contendo um conjunto de terminologias (alinhadas com a estrutura FinOps framework), um schema e um conjunto mínimo de requisitos para dados de faturamento. A especificação se destina a suportar casos de uso comuns a uma variedade de profissionais FinOps. Embora ainda esteja nos estágios iniciais de desenvolvimento e adoção, vale a pena acompanhar porque, com a crescente adoção da indústria, o FOCUS tornará mais fácil para plataformas e pessoas usuárias obter uma visão holística dos gastos com nuvem em uma longa lista de provedores de nuvem e SaaS.

34. Gemini Nano

Avalie

O Gemini do Google é uma família de LLMs (modelos de linguagem de grande porte) básicas projetadas para rodar em uma ampla variedade de hardwares, desde data centers até celulares. O **Gemini Nano** foi especificamente otimizado e dimensionado para funcionar em aceleradores de silício móveis. Ele permite recursos como sumarização de texto de alta qualidade, respostas inteligentes contextuais e correção gramatical avançada. Por exemplo, a compreensão de linguagem do Gemini Nano permite ao Pixel 8 Pro resumir o conteúdo no aplicativo Gravador. A execução no dispositivo remove muitos dos problemas de latência e privacidade associados aos sistemas baseados em nuvem e permite que os recursos funcionem sem conexão de rede. O Android AICore simplifica a integração do modelo em aplicativos Android, mas apenas alguns dispositivos são suportados no momento da redação.

35. HyperDX

Avalie

HyperDX é uma plataforma de código-aberto para observabilidade que unifica os três pilares da área: logs, métricas e rastreamento. Com ela, você pode realizar correlações ponta-a-ponta e ir da reprodução de sessões do navegador até logs e traces com poucos cliques. A plataforma utiliza o ClickHouse como armazenamento central para todos os dados de telemetria, escalando para agregar padrões de logs e condensar bilhões de eventos em clusters distintos. Embora existam diversas plataformas de observabilidade, destacamos o HyperDX pela sua experiência unificada para pessoas desenvolvedoras.

36. IcePanel

Avalie

O **IcePanel** facilita a modelagem e o diagramação colaborativa de arquitetura utilizando o modelo C4, que permite que stakeholders técnicos e de negócios aumentem ou diminuam o zoom no nível de detalhes técnicos que necessitam. A ferramenta suporta a modelagem de objetos de arquitetura, onde metadados e conexões podem ser reutilizados entre diagramas, juntamente com a visualização de fluxo entre esses objetos. O versionamento e marcação permitem que os colaboradores modelem diferentes estados da arquitetura (por exemplo, como está versus como deveria ser) e rastreiem classificações definidas pela pessoa usuária de várias partes da arquitetura. Estamos de olho no IcePanel por seu potencial para melhorar a colaboração em arquitetura, especialmente para organizações com arquiteturas complexas. Para uma alternativa que ofereça um suporte melhor para diagramas como código, confira o Structurizr.

37. Langfuse

Avalie

Langfuse é uma plataforma de engenharia para observabilidade, teste e monitoramento de aplicações baseadas em modelos de linguagem de grande porte (LLMs). Seus SDKs suportam Python, JavaScript e TypeScript, OpenAI, LangChain e LiteLLM entre outras linguagens e frameworks. Você pode hospedar a versão de código aberto por conta própria ou usá-la como um serviço de nuvem pago. Nossos times tiveram uma experiência positiva, particularmente na depuração de cadeias complexas de LLMs, análise de completações e monitoramento de métricas chave como custo e latência entre pessoas usuárias, sessões, regiões geográficas, funcionalidades e versões de modelos. Se você busca construir aplicações de LLM baseadas em dados, o Langfuse é uma boa opção a ser considerada.

38. Qdrant

Avalie

Qdrant é um banco de dados vetoriais de código aberto escrito em Rust. Na edição de setembro de 2023 do Radar, falamos sobre pgvector, uma extensão do PostgreSQL para busca vetorial. No entanto, se você precisar dimensionar o banco de dados vetorial horizontalmente em vários nós, recomendamos avaliar o Qdrant. Ele possui suporte integrado de aceleração SIMD (single instruction, multiple data) para melhorar o desempenho da pesquisa e permite associar payloads JSON a vetores.

39. RISC-V para embedded

Avalie

Enquanto a arquitetura Arm continua expandindo seu impacto — atualizamos nossa avaliação do Arm na nuvem nesta edição — o interesse na arquitetura RISC-V, mais recente e menos estabelecida, também cresce. RISC-V não traz inovações revolucionárias em performance ou eficiência — na verdade, seu desempenho por watt é similar ao de Arm, e não chega a competir em performance absoluta. No entanto, possui código aberto, é modular e não está atrelada a uma única companhia. Isso a torna uma proposta atraente para sistemas embarcados, onde o custo de licenciamento de arquiteturas proprietárias é uma preocupação significativa. É por isso também que o campo de **RISC-V para embedded** está amadurecendo, e diversas empresas, incluindo SiFive e espressif, estão oferecendo placas de desenvolvimento e SoCs para uma ampla variedade de aplicações. Microcontroladores e microprocessadores capazes de rodar o kernel Linux já estão disponíveis atualmente, juntamente com a pilha de software e cadeias de ferramentas correspondentes. Estamos de olho nesse espaço e esperamos ver uma maior adoção nos próximos anos.

40. Tigerbeetle

Avalie

Tigerbeetle é um banco de dados distribuído de código aberto voltado para contabilidade financeira. Ao contrário de outros bancos de dados, ele foi projetado como uma máquina de estado específica de domínio, priorizando segurança e performance. O estado de um nó no cluster é replicado em ordem determinística para outros nós por meio do protocolo de consenso Viewstamped Replication. Percebemos muito potencial nas decisões de design por trás do Tigerbeetle para implementar a contabilidade de partidas dobradas com rigorosas garantias de serialização.

41. WebTransport

Avalie

O **WebTransport** é um protocolo construído sobre o HTTP/3 e oferece comunicação bidirecional entre servidores e aplicativos. O WebTransport oferece vários benefícios em relação ao seu predecessor, o WebSockets, incluindo conexões mais rápidas, menor latência e a capacidade de lidar com fluxos de dados confiáveis e ordenados, bem como não ordenados (como UDP). Ele pode lidar com vários fluxos na mesma conexão sem bloqueio de inicio de linha, permitindo uma comunicação mais eficiente em aplicativos complexos. No geral, o WebTransport é adequado para uma ampla variedade de casos de uso, incluindo aplicativos web em tempo real, streaming de mídia e comunicação de dados da Internet das Coisas (IoT). Embora o WebTransport ainda esteja nos estágios iniciais — o suporte entre navegadores está gradualmente amadurecendo, com bibliotecas populares como o socket. io adicionando suporte para WebTransport — nossas equipes estão atualmente avaliando seu potencial para aplicativos IoT em tempo real.

42. Zarf

Avalie

Zarf é um gerenciador de pacotes declarativo para ambientes Kubernetes offline e semi-conectados. Com o Zarf, você pode construir e configurar aplicações enquanto estiver conectado à internet; uma vez criadas, é possível empacotar e enviar para um ambiente desconectado para implantação. Como uma ferramenta autônoma, o Zarf oferece diversos recursos úteis, incluindo a geração automática de Lista de Materiais de Software (SBOM), registro Docker integrado, painéis Gitea e K9s para gerenciar clusters a partir do terminal. A entrega de software em ambientes isolados (air-gap) para aplicações em nuvem híbrida possui seus desafios; o Zarf soluciona a maior parte deles.

43. ZITADEL

Avalie

ZITADEL é uma ferramenta open-source de gerenciamento de identidade e pessoa usuária, e uma alternativa ao Keycloak. É leve (escrito em Golang), possui opções flexíveis de implementação e é fácil de configurar e gerenciar. Ele também é multitenant, oferece recursos abrangentes para a construção de sistemas de autenticação seguros e escaláveis, especialmente para aplicativos B2B, e possui recursos de segurança integrados, como autenticação multifator e trilhas de auditoria. Ao usar o ZITADEL, as pessoas desenvolvedoras podem reduzir o tempo de desenvolvimento, melhorar a segurança do aplicativo e alcançar escalabilidade para bases de usuários em crescimento. Se você está procurando uma ferramenta amigável, segura e open-source para gerenciamento de pessoas usuárias, o ZITADEL é um forte candidato.

Ferramentas

Adote

44. Conan
45. Kaniko
46. Karpenter

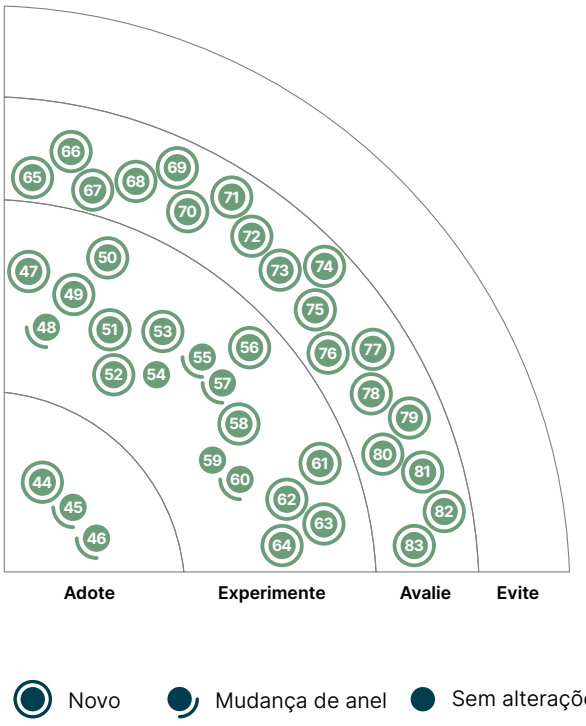
Experimente

47. 42Crunch API Conformance Scan
48. actions-runner-controller
49. Contêiner do Emulador Android
50. AWS CUDOS
51. aws-nuke
52. Bruno
53. Develocity
54. GitHub Copilot
55. Gradio
56. Catálogo de versões do Gradle
57. Maestro
58. Ferramenta Microsoft SBOM
59. Open Policy Agent (OPA)
60. Runner auto-hospedado para GitHub Actions da Philips
61. Pop
62. Renovate
63. Terrascan
64. Velero

Avalie

65. aider
66. Akvorado
67. Baichuan 2
68. Cargo Lambda
69. Codium AI
70. Continue
71. Fern Docs
72. Granted
73. LinearB
74. LLaVA
75. Marimo
76. Mixtral
77. NeMo Guardrails
78. Ollama
79. OpenTofu
80. QAnything
81. System Initiative
82. Tetragon
83. Winglang

Evite



44. Conan

Adote

Conan é uma ferramenta de código aberto para o gerenciamento de dependências para aplicações C/C++. Ele oferece uma interface intuitiva para definir, buscar e gerenciar dependências, facilitando a integração de bibliotecas de terceiros em projetos. O Conan funciona em todos os principais sistemas operacionais e pode ser direcionado a diversas plataformas, incluindo servidores, desktops, dispositivos móveis e embarcados. Ele também pode ser usado para construir e publicar bibliotecas e pacotes C/C++. Os pacotes podem ser compartilhados entre equipes por meio de servidores JFrog Artifactory. Ao aproveitar binários pré-construídos, o Conan reduz significativamente o tempo de compilação, especialmente para dependências pesadas. Ele se integra a sistemas de compilação populares como o CMake e também possui um SDK Python para estender o sistema de compilação para tarefas como assinatura de código. Em nossa experiência, o Conan se traduz em melhor reprodutibilidade de compilação em diferentes ambientes e ciclos de desenvolvimento mais rápidos. As bases de código resultantes ficam mais limpas e fáceis de manter, o que é uma grande vantagem para projetos C e C++ de larga escala. Se você está lutando com o gerenciamento de dependências em seus projetos, o Conan é uma ferramenta imprescindível para aumentar a eficiência do seu desenvolvimento.

45. Kaniko

Adote

Em outubro de 2022, adicionamos o **Kaniko** ao Radar, pouco depois do Kubernetes ter removido o suporte ao Docker. Na época, destacamos a tendência de se afastar do Docker como padrão para a construção de imagens de container dentro de pipelines baseados em container. Desde então, expandimos nossa experiência com Kaniko em diferentes ferramentas e configurações de pipelines. As equipes gostam da sua flexibilidade e performance, por isso estamos movendo o Kaniko para a categoria Adote, destacando-o como a ferramenta padrão nesse espaço.

46. Karpenter

Adote

Uma das funcionalidades fundamentais do Kubernetes é o autoscaling horizontal: a capacidade que permite provisionar novos pods quando demanda por capacidade aumenta e removê-los quando a carga diminui. No entanto, isso só funciona se os nós necessários para hospedar os pods já existirem. O Cluster Autoscaler pode realizar uma expansão básica do cluster acionada por falhas de pods, mas possui flexibilidade limitada. O **Karpenter,** por outro lado, é um autoescalador de nós, mais inteligente, de código aberto Kubernetes Operator. Ele analisa os workloads atuais e restrições de agendamento de pods, seleciona um tipo de instância apropriada e então a inicia ou interrompe conforme a necessidade. O Karpenter é um operador no mesmo espírito de ferramentas como o Crossplane que podem provisionar recursos de cloud fora do cluster. Embora o Karpenter tenha sido originalmente desenvolvido pela AWS para o EKS, ele está se tornando a escolha padrão para provisionadores automáticos de nós entre os provedores de serviços Kubernetes em nuvem. O Azure recentemente iniciou o suporte ao Karpenter com o AKS Karpenter Provider.

47. 42Crunch API Conformance Scan

Experimente

42Crunch API Conformance Scan é uma ferramenta de teste dinâmico projetada para identificar discrepâncias entre o comportamento documentado da sua API e sua implementação real. Ela analisa a definição da especificação da API no formato OpenAPI, que descreve as funcionalidades e respostas esperadas, e a compara com o comportamento real da API. Ao gerar tráfego real e interagir com endpoints ativos, a ferramenta pode identificar quaisquer discrepâncias entre o que a API promete e o que ela entrega. Isso se traduz em diversos benefícios para as equipes de desenvolvimento. Por exemplo, ela captura inconsistências no início do desenvolvimento, economizando tempo e evitando que problemas cheguem à produção. A ferramenta também auxilia na melhoria da qualidade e segurança da API, identificando potenciais vulnerabilidades decorrentes de desvios do comportamento documentado. De forma geral, o API Conformance Scan ajuda a avaliar a postura de segurança de suas APIs identificando problemas como protocolos de autenticação fracos, práticas inseguras de manipulação de dados e validação de entrada insuficiente. A ferramenta fornece relatórios detalhados destacando quaisquer problemas encontrados, juntamente com recomendações para correção.

48. actions-runner-controller

Experimente

actions-runner-controller é um controlador do Kubernetes que opera runners auto-hospedados para o GitHub Actions. Runners auto-hospedados são úteis em cenários onde a tarefa executada pelo GitHub Actions precisa acessar recursos que não estão acessíveis aos runners em nuvem do GitHub ou possuem requisitos específicos de sistema operacional e ambiente diferentes dos fornecidos pelo GitHub. Nesses cenários, onde a equipe usa clusters Kubernetes, o actions-runner-controller orquestra e dimensiona esses runners. Nossas equipes gostam de sua capacidade de dimensionar runners com base no número de workflows em execução em um determinado repositório, organização, enterprise ou cluster Kubernetes, bem como sua capacidade de lidar com runners Linux e Windows.

49. Contêiner do Emulador Android

Experimente

O **Contêiner do Emulador Android** simplifica o processo de teste de aplicativos Android eliminando as complexidades decorrentes de problemas de compatibilidade de SO e dependências do sistema, bem como da configuração de emuladores para várias versões do Android. Tradicionalmente, essa complexidade exigia esforços extras ou levava as equipes a abandonarem completamente os testes automatizados, o que, por sua vez, resultava em ciclos de desenvolvimento e testes mais lentos. O Contêiner Emulador do Android simplifica esse processo, permitindo a integração perfeita em pipelines de CI para testes automatizados. Nossas equipes utilizam esses contêineres principalmente para testes instrumentados, que são executados automaticamente a cada commit para fornecer feedback instantâneo aos desenvolvedores. Além disso, também aproveitamos o Android Emulator Containers para executar testes noturnos completos de ponta a ponta.

50. AWS CUDOS

Experimente

Sempre recomendamos o monitoramento de custos como uma função de aptidão arquitetural. Provedores de nuvem oferecem diversos serviços de monitoramento de gastos, como o AWS Cost Explorer ou o Google Cloud FinOps Hub. No ecossistema AWS, nossos times utilizam os painéis do CUDOS (Cost and Usage Dashboards Operations Solution) para monitorar os gastos do AWS Marketplace segregados por departamentos de negócio ou entidades legais em uma grande organização-mãe. Este painel oferece detalhes abrangentes de custo e uso, com granularidade no nível do recurso, auxiliando na otimização de custos, no acompanhamento de metas de utilização e na obtenção de excelência operacional.

51. aws-nuke

Experimente

aws-nuke é uma ferramenta de código aberto que aborda o desafio comum do acúmulo de recursos não utilizados em contas de desenvolvimento e sandbox da AWS, o que pode levar a ineficiências de custo. A ferramenta identifica e remove todos os recursos deletados dentro de uma conta ou região da AWS, com exceção dos recursos padrão ou gerenciados pela AWS, essencialmente redefinindo o ambiente para um estado inicial. Ela também oferece políticas de exclusão personalizáveis para garantir que os recursos críticos permaneçam protegidos. Utilizamos esta ferramenta tanto para o caso de uso padrão de otimização de custos, quanto em contextos de recuperação de desastres (DR) com bons resultados. Ao automatizar a limpeza em ambientes de desenvolvimento e sandbox, o aws-nuke ajuda a minimizar gastos desnecessários com recursos. Também facilita a desmontagem eficiente da infraestrutura temporária de DR após simulações ou exercícios. Embora estável, o aws-nuke é uma ferramenta muito destrutiva e não se destina ao uso em ambientes de produção. Sempre execute uma simulação para confirmar que os recursos essenciais não serão excluídos.

52. Bruno

Experimente

Bruno é uma alternativa de código aberto para desktop às ferramentas de teste, desenvolvimento e depuração de APIs como Postman e Insomnia. Ele armazena suas coleções localmente no sistema de arquivos, permitindo o uso do Git ou do controle de versão de sua preferência para colaboração. Várias equipes da Thoughtworks estão usando o Bruno e gostam do seu design simples e offline.

53. Develocity

Experimente

Develocity (anteriormente conhecida como Gradle Enterprise) aborda o problema dos ciclos longos de build e teste em projetos de software de larga escala. Ela emprega melhorias de performance, como cache de build e seleção preditiva de testes, para reduzir os ciclos de feedback do desenvolvedor em ambientes locais e de CI/CD. Nossas equipes de plataforma descobriram que ela é útil para acelerar builds e testes, analisar comandos para determinar qual parte do workflow ainda precisa ser otimizada, identificar e solucionar problemas em testes instáveis e realizar análises no hardware usado para executá-los.

54. GitHub Copilot

Experimente

Embora o mercado de assistentes de programação por IA esteja cada vez mais movimentado, o **GitHub Copilot** continua sendo nossa escolha padrão e é utilizado por muitas de nossas equipes. Desde a última vez que escrevemos sobre o GitHub Copilot, as melhorias mais interessantes vieram no recurso de chat. Por exemplo, não é mais necessário poluir o código com comentários como comandos; agora, um chat integrado ajuda você a dar comandos sem escrever um comentário. O chat integrado também pode alterar o código, não apenas escrever novas linhas. Agora você também pode expandir significativamente o contexto do chat ao fazer perguntas sobre seu código, usando a tag @ workspace. Isso permite que você faça perguntas sobre toda a base de código, não apenas sobre os arquivos abertos. Você pode expandir ainda mais esse contexto com a versão Copilot Enterprise, que obtém contexto de todos os repositórios que você hospeda no GitHub. Por fim, o GitHub começou a direcionar algumas solicitações de chat para um modelo GPT-4 mais poderoso, e a disponibilidade do chat nas populares IDEs da JetBrains é iminente (embora ainda em beta privado no momento da redação). Esses lançamentos mostram que o ritmo de melhorias na área não desacelerou. Se você experimentou um assistente de programação no ano passado e o descartou, recomendamos que continue monitorando os recursos sendo lançados e dê outra chance.

55. Gradio

Experimente

Gradio é uma biblioteca Python de código aberto que facilita a criação de interfaces web interativas para modelos de Aprendizado de Máquina (ML). Ao prover uma interface gráfica sobre os modelos, o Gradio permite que as pessoas sem conhecimento técnico compreendam melhor os dados de entrada, restrições e saídas. A biblioteca ganhou bastante força na área de Inteligência Artificial Generativa, pois torna modelos generativos facilmente acessíveis para experimentação. Normalmente, só adicionamos tecnologias em Experimente após observarmos seu uso em produção. O foco e força do Gradio são a experimentação e prototipagem, e já o utilizamos diversas vezes para esse fim. Recentemente, uma de nossas equipes utilizou o Gradio para auxiliar uma cliente em demonstrações ao vivo durante grandes eventos. Estamos muito satisfeitos com as capacidades do Gradio para esses casos de uso, o que motivou a inclusão em Experimente.

56. Catálogo de versões do Gradle

Experimente

O **catálogo de versões do Gradle** é um recurso útil da ferramenta de compilação Gradle que permite o gerenciamento centralizado de dependências no arquivo de compilação. Nossas equipes têm o considerado especialmente valioso para projetos Android multi-módulo. Ao invés de programar nomes e versões de dependências em arquivos de compilação individuais e gerenciar atualizações manualmente, você pode criar um catálogo central de versões dessas dependências e então referenciá-lo de forma segura por tipo com a assistência do Android Studio.

57. Maestro

Experimente

Maestro é uma ferramenta extremamente útil para testar fluxos complexos em aplicativos mobile. Nossos times descobriram que ele é fácil de aprender, entender e integrar ao fluxo de desenvolvimento. O Maestro suporta diversas plataformas mobile, incluindo iOS, Android, aplicativos React Native e Flutter. Sua sintaxe declarativa em YAML simplifica a automatização de interações complexas de UI mobile (interface da pessoa usuária para dispositivos móveis). Baseado na evolução da ferramenta, marcada por recursos aprimorados como suporte abrangente ao iOS e a introdução de ferramentas como Maestro Studio e o Maestro Cloud, incentivamos que qualquer pessoa que busque otimizar seus processos de teste de aplicativos mobile experimente o Maestro.

58. Microsoft SBOM tool

Experimente

A **Microsoft SBOM tool** é uma ferramenta de código aberto para gerar Listas de Materiais de Software (SBOM) compatíveis com SPDX. Já falamos anteriormente sobre a necessidade de SBOM, e esta ferramenta facilita o início do processo. A ferramenta SBOM suporta diversos gerenciadores de pacotes populares (incluindo npm, pip e Gradle), tornando-a compatível com uma ampla variedade de projetos. É muito fácil de usar e pode ser integrada aos fluxos de desenvolvimento existentes, incluindo a integração com pipelines de CI/CD. Ao aproveitar o SBOM gerado com esta ferramenta, as pessoas desenvolvedoras obtêm várias vantagens. A segurança aprimorada do software é um benefício fundamental, pois uma visão clara dos componentes permite uma identificação mais fácil de vulnerabilidades e um melhor gerenciamento de riscos. A conformidade com licenças também é aprimorada, pois as pessoas desenvolvedoras podem garantir a adesão a todos os contratos relevantes. Além disso, o SBOM promove transparência na cadeia de fornecimento de software, auxiliando no rastreamento de dependências e mitigando riscos potenciais. Se você deseja agilizar a geração de SBOM, melhorar a segurança do software e obter controle sobre sua cadeia de fornecimento de software, considere experimentar a ferramenta Microsoft SBOM.

59. Open Policy Agent (OPA)

Experimente

Open Policy Agent (OPA) é uma estrutura e linguagem unificada para declarar, impor e controlar políticas. Para nossos times, ele se tornou a forma favorita de definir políticas para sistemas distribuídos, especialmente quando precisamos implementar conformidade no ponto de mudança. O OPA permite que os times implementem diversos padrões de engenharia de plataforma, como controlar o que é implantado em clusters Kubernetes, reforçar o controle de acesso entre serviços em uma malha de serviços e implementar políticas de segurança como código granulares para acessar recursos de aplicativos. Embora existam algumas complexidades associadas às implementações do OPA, ele provou ser uma ferramenta altamente valiosa para garantir a conformidade em uma cultura DevOps. Também continuamos atentos à extensão e maturidade do OPA para soluções além de sistemas operacionais, alcançando soluções centradas em dados (big data).

60. Runner auto-hospedado para GitHub Actions da Philips

Experimente

Embora os runners do GitHub Actions cubram uma ampla variedade dos ambientes de execução mais comuns e sejam os mais rápidos para começar, às vezes as equipes precisam gerenciar runners auto-hospedados. Isso acontece, por exemplo, quando a política da organização permite apenas implantações em uma infraestrutura privada hospedada dentro do próprio perímetro de segurança da organização. Nesses casos, as equipes podem usar o **runner auto-hospedado para GitHub Actions da Philips**, um módulo Terraform que provisiona runners customizados em instâncias spot da AWS EC2. O módulo também cria um conjunto de Lambdas que lida com o gerenciamento de ciclo de vida (aumento e redução de escala) desses runners. Em nossa experiência, esta ferramenta simplifica muito o provisionamento e gerenciamento de runners auto-hospedados do GitHub Actions. Uma alternativa para equipes que usam Kubernetes é o actions-runner-controller.

61. Pop

Experimente

A programação em pares continua sendo uma técnica essencial para nós, pois ajuda a melhorar a qualidade do código e a disseminar o conhecimento dentro da equipe. Embora seja melhor feita presencialmente, nossas equipes distribuídas tem explorado ferramentas, para tornar o pareamento remoto o mais agradável e eficaz possível, como Tuple, Visual Studio Live Share, Code With Me e ferramentas gerais de chat e conferência. A última ferramenta que chamou a nossa atenção foi a **Pop** (anteriormente conhecida como Screen). Vinda dos fundadores da Screenhero, ela suporta compartilhamento de tela com várias pessoas, anotações e áudio/vídeo de alta qualidade. Algumas de nossas equipes a utilizaram extensivamente para programação em pares e sessões de trabalho remoto, e relataram uma experiência positiva.

62. Renovate

Experimente

O monitoramento e a atualização automáticos de dependências como parte do processo de compilação de software se tornaram prática padrão em todo o setor. Isso elimina as incertezas de se manter atualizado com as atualizações de segurança para pacotes de código aberto assim que são lançadas. Por muitos anos, o Dependabot foi a ferramenta padrão para essa prática, mas o **Renovate** se tornou a ferramenta preferida para muitas de nossas equipes. Elas consideram o Renovate mais adequado ao ambiente moderno de desenvolvimento de software, onde um sistema implantável depende não apenas de código e bibliotecas, mas também abrange ferramentas de tempo de execução, infraestrutura e serviços de terceiros. O Renovate cobre dependências desses artefatos auxiliares, além do código. Nossas equipes também descobriram que o Renovate oferece mais flexibilidade por meio de opções de configuração e personalização. Embora o Dependabot permaneça como uma opção padrão segura e esteja convenientemente integrado ao GitHub, recomendamos avaliar o Renovate para ver se ele pode reduzir ainda mais o encargo manual das pessoas desenvolvedoras para manter seus ecossistemas de aplicativos seguros e protegidos.

63. Terrascan

Experimente

Terrascan é um analisador estático de código para infraestrutura como código (IaC) projetado para detectar vulnerabilidades de segurança e problemas de compliance antes do provisionamento de infraestrutura nativa de nuvem. Ele suporta varredura de Terraform: Terraform, Kubernetes (JSON/YAML), Helm, AWS CloudFormation, Azure Resource Manager, Dockerfiles e GitHub. O pacote de políticas padrão cobre todos os principais provedores de nuvem, GitHub, Docker e Kubernetes. Nossas equipes utilizam o Terrascan localmente como um hook de pre-commit e o integram em pipelines de CI/CD para detectar vulnerabilidades e violações de IaC.

64. Velero

Experimente

Velero é uma ferramenta de código aberto para backup e restauração de recursos e volumes persistentes do Kubernetes. Ela simplifica a recuperação de desastres e migrações de clusters permitindo backups agendados e sob demanda. O Velero também possibilita controles mais granulares sobre quais recursos serão copiados e sobre o fluxo de trabalho de backup/restauração (com hooks). Nossas equipes gostam da facilidade de uso da ferramenta e sua dependência nas APIs do Kubernetes em vez de camadas de nível inferior, como o etcd.

65. aider

Avalie

O **aider** é um assistente de programação por IA de código aberto. Assim como muitas ferramentas de código aberto nessa área, o aider não possui integração direta com IDE, sendo iniciado como uma CLI no terminal. O aider é interessante porque fornece uma interface de chat com acesso de escrita para a base de código em vários arquivos, enquanto muitos dos produtos de assistente de codificação atuais lêem apenas o código ou podem alterar apenas um arquivo por vez. Isso permite que o aider auxilie na implementação de conceitos que se estendem por vários arquivos (por exemplo,adicionar localizadores ao meu HTML e também usá-los no meu teste funcional) e na criação de novos arquivos e estruturas de pastas na base de código (por exemplo,criar um novo componente semelhante ao da pasta X). Como o aider é open-source e não um produto hospedado, você precisa ter sua própria chave de API OpenAI ou Azure OpenAI para usá-lo. Por um lado, isso pode ser ótimo para uso ocasional, pois você só paga pelo uso. Por outro lado, o aider parece ser bastante falador em suas interações com a API de IA, portanto, fique atento aos custos de solicitação e limites de taxa ao usá-lo.

66. Akvorado

Avalie

Akvorado é uma ferramenta de código aberto para o monitoramento e análise de rede. Ele captura fluxos de rede, como Netflow/IPFIX e sFlow, enriquece-os com nomes de interfaces e informações geográficas e, em seguida, salva os fluxos atualizados no ClickHouse para análises futuras. Embora o OpenTelemetry esteja ganhando adoção para observar o tráfego em nível de aplicativo, muitas vezes encontramos desafios na camada de rede que podem ser difíceis de identificar e solucionar. Ferramentas como o Akvorado são bastante úteis nessas situações, pois ajudam a analisar os fluxos de rede em vários dispositivos na topologia da rede.

67. Baichuan 2

Avalie

O **Baichuan 2** faz parte de uma nova geração de modelos de linguagem ampla de código aberto. Treinado em um corpus de alta qualidade com 2,6 trilhões de tokens, alcançou um desempenho muito bom para o seu tamanho em benchmarks de chinês, inglês e multilinguagem. Além disso, o Baichuan 2 foi treinado em diversos corpora específicos de domínio, incluindo conjuntos de dados de saúde e direito, por isso preferimos utilizá-lo nessas e em áreas relacionadas.

68. Cargo Lambda

Avalie

A eficiência e performance do Rust o tornam uma boa opção para a computação sem servidor. Outra vantagem é que as funções Rust não necessitam de um runtime, resultando em tempos de inicialização rápidos. No entanto, a experiência da pessoa desenvolvedora para escrever funções em Rust não era excelente. Isso mudou com o **Cargo Lambda**. Como um subcomando do Cargo, ele se integra ao workflow típico do Rust e permite executar e testar suas funções AWS Lambda na máquina da pessoa desenvolvedora sem a necessidade de Docker, VMs ou outras ferramentas. Usando uma cadeia de ferramentas Zig, o Cargo Lambda pode realizar a compilação cruzada das funções em diversos sistemas operacionais para os sandboxes Linux usados pelo AWS Lambda, sendo compatível com arquiteturas Arm e Intel como alvos.

69. Codium AI

Avalie

No movimentado espaço emergente de assistentes de programação por IA, alguns produtos adotam uma abordagem mais focada, em vez de competir com os líderes de mercado consolidados. O **Codium AI** foca na geração de testes com IA. Ele funciona para todas as linguagens, mas possui suporte avançado para stacks comuns, como JavaScript e Python. Gostamos particularmente do fato de que a ferramenta, em vez de levar as pessoas desenvolvedoras diretamente para o código de teste, oferece uma lista de descrições de cenários em linguagem natural para revisão. Isso facilita o raciocínio sobre os cenários e a decisão de quais transformar em código de teste. Para melhorar ainda mais a geração de testes para uma codebase e caso de uso específicos, as pessoas usuárias podem fornecer testes de exemplo e instruções gerais para aprimorar as informações usadas pela IA para gerar os testes.

70. Continue

Avalie

O **Continue** é um assistente de programação open-source para VS Code e JetBrains IDEs. Percebemos muito potencial nele, já que ele elimina a necessidade de copiar e colar código entre uma interface de chat e grandes modelos de linguagem (LLMs) graças à integração direta na IDE. Ele suporta diversos modelos comerciais e de código aberto e facilita a experimentação com diferentes provedores de LLM, incluindo LLMs auto-hospedados. É possível até rodar o Continue sem conexão com a internet.

71. Fern Docs

Avalie

Uma característica marcante de APIs REST bem-sucedidas é a documentação completa de seus contratos. Pessoas desenvolvedoras tendem a adotar e utilizar mais prontamente APIs cujo comportamento e sintaxe estejam descritos com precisão e de forma organizada. Manter essa documentação atualizada à medida que o contrato evolui pode ser demorado e é uma tarefa facilmente negligenciada. O **Fern Docs** auxilia nesse processo reduzindo o trabalho manual envolvido na escrita e manutenção da documentação de APIs. O Fern gera automaticamente um website com documentação atraente e utilizável a partir de um arquivo de especificação que pode ser versionado junto com o código da API. Embora nossas impressões iniciais sobre este produto sejam positivas, o Fern requer a manutenção das informações da API em um arquivo de configuração proprietário. Embora ele forneça uma maneira de converter especificações OpenAPI em seu próprio formato de configuração, preferiríamos uma ferramenta que gere documentos diretamente do código fonte anotado.

72. Granted

Avalie

Como a estratégia de múltiplas contas é comum em ambientes AWS de organizações, as pessoas engenheiras frequentemente precisam alternar entre várias contas em um curto período. O **Granted** é uma ferramenta de linha de comando que simplifica a abertura simultânea de várias contas no navegador, facilitando a troca de contas. Ele aproveita os recursos nativos de cada navegador para isolar identidades múltiplas, utilizando os Contêineres Multiconta do Firefox e os Perfis do Chromium. Se um serviço específico (como S3) for especificado como argumento, o Granted abrirá a página de destino do serviço. Atualmente, o Granted oferece suporte apenas para AWS. Vale ressaltar que ele armazena as credenciais temporárias do AWS SSO com segurança em um chaveiro, em vez de texto simples no disco.

73. LinearB

Avalie

O **LinearB** é uma plataforma projetada para capacitar líderes de engenharia com insights baseados em dados para melhoria contínua. Ele aborda três áreas principais: benchmarking, automação de workflow e investimento. Nossa experiência com a ferramenta de métricas do LinearB destaca seu potencial para apoiar uma cultura de melhoria contínua. Uma de nossas equipes utilizou a plataforma para rastrear métricas de engenharia, identificar e discutir oportunidades de melhoria e definir etapas acionáveis baseadas em dados, levando a um progresso mensurável. Ficamos satisfeitos em ver que isso se alinha à proposta de valor central do LinearB: avaliar, automatizar e aprimorar. O LinearB integra-se com GitHub, GitLab, Bitbucket e Jira. Ele oferece um conjunto abrangente de métricas de engenharia pré-configuradas, com forte foco nas métricas DORA (frequência de deploy, lead time, taxa de falha de alterações e tempo de restauração). Como fortes defensores das quatro métricas-chave definidas pela pesquisa DORA, gostamos da ênfase do LinearB em medir o que realmente importa para o desempenho da entrega de software. Historicamente, coletar métricas específicas de DORA tem sido um desafio. As equipes recorriam a instrumentação complexa de pipelines de CD, painéis personalizados ou processos manuais. Embora nossa experiência seja limitada a uma equipe, o LinearB parece ser uma alternativa interessante para a coleta e monitoramento de métricas de engenharia, além de fomentar uma abordagem baseada em dados para a melhoria contínua.

74. LLaVA

Avalie

O **LLaVA (Large Language and Vision Assistant - Assistente de Linguagem e Visão Ampla)** é um modelo multimodal de grande porte e código aberto que conecta um codificador de visão e um modelo de linguagem de grande porte (LLM) para compreensão visual e linguística de uso geral. A alta proficiência do LLaVA em seguir instruções o posiciona como um forte concorrente entre os modelos de Inteligência Artificial multimodais. A versão mais recente, o LLaVA-NeXT, permite melhor desempenho em tarefas de resposta a perguntas. Entre os modelos open-source para auxílio de linguagem e visão, o LLaVA é uma opção promissora ao ser comparado com o GPT-4 Vision. Nossas equipes têm experimentado o LLaVA para responder a perguntas visuais.

75. Marimo

Avalie

Marimo oferece uma nova abordagem para notebooks Python, priorizando a reprodutibilidade e a interatividade. Ela soluciona os desafios relacionados ao estado oculto em notebooks tradicionais (como o Jupyter) que podem levar a comportamentos inesperados e dificultar a reprodutibilidade. Isso é feito armazenando os notebooks como arquivos Python simples, sem estado oculto, e usando uma ordem de execução determinística baseada em dependências (quando uma variável muda, todas as células afetadas são executadas automaticamente). O Marimo também vem com elementos de interface do usuário interativos que propagam da mesma forma as alterações de valor para as células que dependem delas. Como pode ser implantado como um aplicativo web, também é uma ferramenta útil para demonstrações e prototipagem. Embora estejamos animadas com o potencial do Marimo, particularmente em termos de reprodutibilidade para exploração e análise de dados, continuamos a alertar contra o uso de notebooks em produção.

76. Mixtral

Avalie

Mixtral faz parte da família de modelos de linguagem de grande porte aberta Mistral recém-lançada, que utiliza a arquitetura sparse Mixture of Experts. A família de modelos está disponível tanto na forma pré-treinada quanto fine-tuned, com tamanhos de parâmetros de 7B e 8×7B. Seu tamanho, natureza de pesos abertos, desempenho em benchmarks e comprimento de contexto de 32.000 tokens o tornam uma boa opção para LLMs auto-hospedados. É importante destacar que esses modelos de pesos abertos não são ajustados para segurança por padrão, e as pessoas usuárias precisam refinar a moderação com base em seus próprios casos de uso. Temos experiência com essa família de modelos no desenvolvimento do Aalap.1-bf16, um modelo Mistral 7B ajustado e treinado em dados relacionados a tarefas jurídicas indianas específicas, que teve um desempenho bastante satisfatório em comparação a um custo acessível.

77. NeMo Guardrails

Assess

NeMo Guardrails é um kit de ferramentas de código aberto fácil de usar da NVIDIA que permite que pessoas desenvolvedoras implementem guardas para modelos de linguagem de grande porte (LLMs) utilizados em aplicações conversacionais. Embora os LLMs tenham um imenso potencial para construir experiências interativas, suas limitações inerentes em relação à precisão factual, vieses e potencial uso indevido exigem salvaguardas. Os guardrails oferecem uma abordagem promissora para garantir LLMs responsáveis e confiáveis. Apesar de haver outras opções de guardrails para LLM, nossas equipes consideram o NeMo Guardrails particularmente útil porque ele suporta regras programáveis, integração em tempo de execução e pode ser aplicado a aplicações LLM existentes sem extensas modificações no código.

78. Ollama

Assess

Ollama é uma ferramenta código aberto para executar e gerenciar modelos de linguagem de grande porte (LLMs) na sua máquina local. Anteriormente, discutimos os benefícios dos LLMs auto-hospedados, e é animador ver o ecossistema amadurecer com ferramentas como o Ollama. Ele suporta diversos modelos populares — incluindo LLaMA-2, CodeLLaMA, Falcon e Mistral — que você pode baixar e executar localmente. Uma vez baixado, é possível interagir com o modelo e executar tarefas através da CLI, API ou SDK. Estamos avaliando o Ollama e observando sucessos iniciais, pois ele melhora a experiência das pessoas desenvolvedoras ao trabalhar com LLMs localmente.

79. OpenTofu

Assess

O **OpenTofu** é um fork do Terraform criado em resposta a uma recente mudança ambígua na licença pela HashiCorp. É um projeto de código aberto aceito pela Linux Foundation e apoiado por diversas organizações, incluindo fornecedores terceirizados. A versão atual é compatível com a última versão de código aberto do Terraform, e a versão 1.7 adiciona criptografia no lado da cliente. O futuro do OpenTofu é incerto em relação ao grau de compatibilidade que manterá com versões futuras do Terraform. Também há dúvidas sobre o suporte de longo prazo por seus apoiadores atuais. Recomendamos ficar de olho no projeto, mas adotamos uma postura cautelosa em relação ao uso, exceto para equipes com capacidade para gerenciar riscos, incluindo a possibilidade de contribuição para a base de código.

80. QAnything

Assess

Os modelos de linguagens de grande porte (LLMs) e as técnicas de geração aumentada por recuperação (RAG) aprimoraram muito nossa capacidade de sintetizar e extrair informações. Estamos observando ferramentas emergentes estão aproveitando esse avanço, e o QAnything é um exemplo. O **QAnything** é um motor de gerenciamento de conhecimento com interface de perguntas e respostas, capaz de resumir

e extrair informações de diversos formatos de arquivo, incluindo PDF, DOCX, PPTX, XLSX e MD, entre outros. Para questões relacionadas a segurança de dados, o QAnything também suporta instalação offline. Algumas de nossas equipes utilizam o QAnything para construir sua base de conhecimento interna. Em cenários de Inteligência Artificial Generativa (GenAI) com maior profundidade setorial (como geração de resumos para relatórios de investimento), também tentamos usar essa ferramenta para provas de conceito antes de construir produtos reais, demonstrando o potencial de LLMs e RAG.

81. System Initiative

Assess

Poucas coisas tem surgido nos últimos anos para desafiar o domínio do Terraform como uma ferramenta de infraestrutura como código. Embora alternativas como Pulumi, CDK e, mais recentemente, Wing tenham surgido, o paradigma modular e declarativo do Terraform provou ser o mais duradouro. De fato, todas essas abordagens compartilham o objetivo comum de código modular para criar infraestrutura monolítica. **System Initiative** é uma nova ferramenta experimental que representa uma direção radicalmente nova para o trabalho de DevOps. Uma maneira de ver o System Initiative é como um gêmeo digital para sua infraestrutura. Alterações interativas no estado do System Initiative resultam em conjuntos de alterações correspondentes que podem ser aplicados à própria infraestrutura. Da mesma forma, as alterações na infraestrutura são refletidas no estado do System Initiative. Uma das grandes vantagens dessa abordagem é o ambiente colaborativo que ele cria para coisas como implantação de aplicativos e observabilidade. Os engenheiros interagem com o System Initiative por meio de uma interface do usuário que possui uma representação gráfica de todo o ambiente. Além de gerenciar a infraestrutura em nuvem, você também pode usar a ferramenta para gerenciar contêineres, scripts, ferramentas e muito mais. Embora sejamos geralmente céticos em relação a esse tipo de ferramenta de interface gráfica, o System Initiative pode ser estendido para lidar com novos ativos ou impor políticas por meio de código TypeScript. Gostamos muito do pensamento criativo empregado nesta ferramenta e esperamos que incentive outros a romper com o status quo das abordagens de infraestrutura como código. O System Initiative é gratuito e open source sob a licença Apache 2.0, e atualmente está em beta aberto. Os próprios mantenedores ainda não recomendam a ferramenta para uso em produção, mas achamos que vale a pena conferir em seu estado atual para experimentar uma abordagem completamente diferente para ferramentas DevOps.

82. Tetragon

Assess

Tetragon é uma ferramenta de código aberto para observabilidade e aplicação de regras de segurança em tempo de execução baseada em eBPF. Já mencionamos o Falco para detecção de ameaças à segurança em edições anteriores do Radar. O Tetragon vai além da detecção de ameaças ao aproveitar o eBPF para implementar políticas de segurança em tempo de execução no kernel do Linux. Você pode usar o Tetragon como uma ferramenta independente em bare metal ou dentro do ambiente Kubernetes.

83. Winglang

Assess

Estamos observando bastante movimentação no espaço de Infraestrutura como Código (IaC) com o surgimento de ferramentas como o **Winglang**. O Winglang adota uma abordagem diferenciada para definir infraestrutura e comportamento em tempo de execução. Ele fornece abstrações de alto nível sobre especificidades de plataforma oferecidas por ferramentas como CloudFormation, Terraform, Pulumi e Kubernetes. Com o Winglang, você escreve um código que é executado em tempo de compilação para gerar a configuração da infraestrutura e, em seguida, o código que é executado em tempo de execução para o comportamento da aplicação. Ele oferece um modo de simulação para execução local e possui uma estrutura de teste integrada. Estamos de olho nesta ferramenta interessante; é uma possível prévia da direção futura do IaC.

Linguagens e Frameworks

Adote

—

Experimente

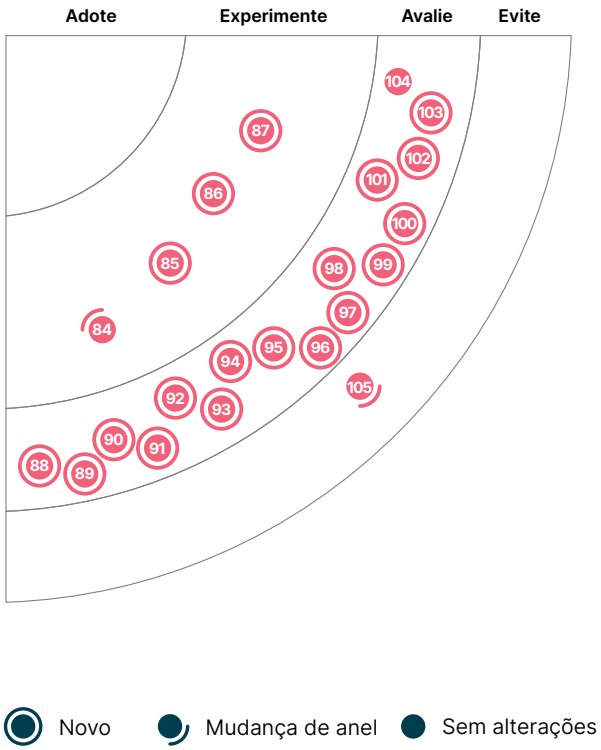
84. Astro
85. DataComPy
86. Pinia
87. Ray

Avalie

88. Android Adaptability
89. Concrete ML
90. Crabviz
91. Crux
92. Databricks Asset Bundles
93. Electric
94. LiteLLM
95. LLaMA-Factory
96. MLX
97. Mojo
98. Otter
99. Pkl
100. Rust para o desenvolvimento de Interface para pessoas usuárias (UI)
101. vLLM
102. Voyager
103. WGPU
104. Zig

Evite

105. LangChain



84. Astro

Experimente

O framework **Astro** está ganhando cada vez mais popularidade na comunidade. Uma de nossas equipes utilizou o Astro para construir sites focados em conteúdo, como blogs e sites de marketing. O Astro é um framework para aplicações multi-página que renderiza HTML no servidor e minimiza a quantidade de JavaScript enviada pela rede. Apesar de incentivar o envio de apenas HTML, gostamos do fato de o Astro permitir, quando apropriado, o uso de componentes ativos escritos na framework JavaScript de front-end da sua escolha. Isso é possível através da sua arquitetura de ilhas (island architecture). As ilhas são regiões interativas dentro de uma única página, onde o JavaScript necessário é baixado somente quando necessário. Dessa forma, a maior parte do site é convertida em HTML estático e rápido, e as partes em JavaScript são otimizadas para carregamento paralelo. Nossa equipe gosta tanto a performance de renderização de página quanto a velocidade de compilação do Astro. A sintaxe de componentes do Astro é uma extensão simples do HTML, tornando a curva de aprendizado bem tranquila.

85. DataComPy

Experimente

Comparar DataFrames é uma tarefa comum na Engenharia de Dados, frequentemente realizada para verificar se não ocorreram desvios ou inconsistências significativas entre os resultados de duas abordagens de transformação de dados. O **DataComPy** é uma biblioteca Python que facilita a comparação de dois DataFrames no pandas, Spark e outras tecnologias. A biblioteca vai além de verificações básicas de igualdade, fornecendo insights detalhados sobre discrepâncias nos níveis de linha e coluna. O DataComPy também permite especificar tolerância absoluta ou relativa para comparação de colunas numéricas, bem como diferenças conhecidas que não precisam ser destacadas em seu relatório. Algumas de nossas equipes o utilizam como parte de seu pacote de smoke testing; elas consideram eficiente para comparar DataFrames grandes e complexos, e seus relatórios são fáceis de entender e acionar.

86. Pinia

Experimente

Pinia é uma biblioteca de store e framework de gerenciamento de estado para Vue.js. Ela utiliza sintaxe declarativa e oferece sua própria API de gerenciamento de estado. Comparado ao Vuex, o Pinia oferece uma API mais simples e enxuta, APIs no estilo Composition e, mais importante, possui suporte sólido a inferência de tipos quando usado com TypeScript. O Pinia é endossado pela equipe Vue.js como uma alternativa confiável ao Vuex e atualmente é a biblioteca oficial de gerenciamento de estado para Vue.js. Nossos times estão utilizando o Pinia por sua simplicidade e facilidade de implementação.

87. Ray

Experimente

As cargas de trabalho de aprendizado de máquina (ML) da atualidade exigem cada vez mais recursos computacionais. Ambientes de desenvolvimento em nodo único, como o seu laptop, apesar de convenientes, não escalam para atender a essas demandas. O **Ray** é um framework unificado para dimensionar código Python e de IA do laptop para clusters. O Ray é essencialmente um framework de computação distribuída bem encapsulado, com uma série de bibliotecas de IA para simplificar o trabalho de ML. Ao se integrar com outras frameworks (por exemplo, PyTorch and TensorFlow), ele pode ser usado para construir plataformas de ML em larga escala. Empresas como OpenAI e Bytedance usam o Ray extensivamente para treinamento e inferência de modelos. Também usamos suas bibliotecas de IA para auxiliar no treinamento distribuído e no ajuste de hiperparâmetros em nossos projetos. Recomendamos que você experimente o Ray ao construir projetos de ML escaláveis.

88. Android Adaptability

Avalie

Alguns aplicativos e jogos mobile podem ser tão exigentes que causam superaquecimento momentâneo em poucos minutos. Nesse estado, a frequência da CPU e GPU é reduzida para auxiliar no resfriamento do dispositivo, mas também resulta em taxas de quadros (frame rates) menores em jogos. Quando a situação térmica melhora, as taxas de quadros aumentam novamente e o ciclo se repete, tornando a experiência desorganizada. O **Android Adaptability**, um novo conjunto de bibliotecas, permite que as pessoas desenvolvedoras de aplicativos respondam a mudanças no desempenho e na situação térmica. O Android Dynamic Performance Framework (ADPF) inclui a Thermal API para fornecer informações sobre o estado térmico e a Hint API para ajudar o Android a escolher o ponto de operação ideal da CPU e o posicionamento do núcleo. As equipes que utilizam Unity vão aproveitar mais o pacote Unity Adaptive Performance útil, pois ele funciona com ambas as APIs.

89. Concrete ML

Avalie

Já discutimos anteriormente a técnica de Criptografia Homomórfica, que permite realizar cálculos diretamente em dados criptografados. **Concrete ML** é uma ferramenta de código aberto que possibilita o aprendizado de máquina com preservação de privacidade. Construída sobre o Concrete, ela simplifica o uso da criptografia totalmente homomórfica (FHE) para pessoas cientistas de dados, auxiliando-os na conversão automática de modelos de aprendizado de máquina em seus equivalentes homeomórficos. Os modelos integrados do Concrete ML possuem APIs quase idênticas às suas contrapartes do scikit-learn. Também é possível converter redes PyTorch para FHE usando as APIs de conversão do Concrete ML. É importante ressaltar que a FHE com Concrete ML pode ser lenta sem o uso de hardware especializado.

90. Crabviz

Avalie

Crabviz é um plugin de código aberto para o Visual Studio Code que permite a criação de diagramas de chamada (call graphs). Esses diagramas são interativos, o que é essencial ao trabalhar com bases de código de tamanho médio, como microsserviços. Eles exibem tipos, métodos, funções e interfaces agrupados por arquivo, além de mostrar as relações de chamadas de função e implementações de interface. Como o Crabviz é baseado no Language Server Protocol, ele suporta qualquer linguagem, desde que o servidor de linguagem correspondente esteja instalado. Isso significa, porém, que o Crabviz está limitado à análise estática de código, o que pode não ser suficiente para alguns casos de uso. O plugin é escrito em Rust e está disponível no Marketplace do Visual Studio Code.

91. Crux

Avalie

O **Crux** é um framework de código aberto para o desenvolvimento de aplicativos multiplataforma escrito em Rust. Inspirado na arquitetura Elm, o Crux organiza a lógica de negócios no core e a camada de interface da pessoa usuária em frameworks nativos como SwiftUI, Jetpack Compose, React/Vue ou frameworks baseados em WebAssembly (como Yew). Com o Crux, você pode escrever código de comportamento livre de efeitos colaterais em Rust e compartilhá-lo entre iOS, Android e a web.

92. Databricks Asset Bundles

Avalie

A recente disponibilização em preview público dos **Databricks Asset Bundles (DABs)** incluídos na versão 0.205 ou superior da Databricks CLI está se tornando a forma oficialmente recomendada para empacotamento de Databircks assets para o controle de versão, testes e implantação. Em nossas equipes, o DABs já começaram a substituir o dbx. Os DABs permitem o empacotamento da configuração de workflows, jobs e tasks, assim como o código a ser executado nessas tasks, como um único bundle que pode ser implantado em múltiplos ambientes. Ele vem com templates para tipos comuns de assets e suporta templates customizados. Embora os DABs incluam templates para notebooks e suportem a implantação deles em produção, continuamos a desaconselhar a produção de notebooks e incentivamos a escrita de código de produção com práticas de engenharia que suportem as necessidades de manutenibilidade, resiliência e escalabilidade desse tipo de carga de trabalho.

93. Electric

Avalie

Electric é um framework de sincronização local-first para aplicações mobile e web. Local-first é um paradigma de desenvolvimento onde o código da sua aplicação se comunica diretamente com um banco de dados local integrado, e a sincronização de dados ocorre em segundo plano através de replicação ativa-ativa para o banco de dados central. Com o Electric, você tem SQLite como opção local integrada e PostgreSQL para o armazenamento central. Embora o local-first melhore muito a experiência da pessoa usuária, ele também traz desafios. Pensando nisso, os inventores do CRDT trabalharam no framework Electric para facilitar o desenvolvimento.

94. LiteLLM

Avalie

LiteLLM é uma biblioteca que permite uma integração descomplicada com APIs de diversos provedores de modelos de linguagem de grande porte (LLMs). Ela padroniza as interações por meio de um formato de API compatível com o OpenAI.O LiteLLM oferece suporte a uma ampla variedade de provedores e modelos e disponibiliza uma interface unificada para funcionalidades de complementação, incorporação e geração de imagens. A biblioteca simplifica a integração ao traduzir entradas para atender aos requisitos específicos de endpoint de cada provedor. Isso é particularmente útil no cenário atual, onde a ausência de especificações padronizadas de API para fornecedores de LLM dificulta a inclusão de vários LLMs em projetos. Nossas equipes utilizaram o LiteLLM para alternar modelos subjacentes em aplicações de LLM, superando um desafio significativo de integração. No entanto, é fundamental reconhecer que as respostas dos modelos a prompts idênticos podem variar, indicando que um método de invocação consistente por si só pode não otimizar totalmente o desempenho da complementação. Vale ressaltar que o LiteLLM possui diversos outros recursos, como o servidor proxy, que não estão no escopo deste blip.

95. LLaMA-Factory

Avalie

Continuamos alertando contra o fine-tune apressado de modelos de linguagem de grande porte (LLMs) a menos que seja absolutamente crítico — isso acarreta custos e demanda alto nível de expertise. Porém, acreditamos que a **LLaMA-Factory** pode ser útil quando o fine-tuning for necessário. É uma estrutura de código aberto e fácil de usar para fine-tuning e treinamento de LLMs. Com suporte para LLaMA, BLOOM, Mistral, Baichuan, Qwen e ChatGLM, torna o conceito complexo de fine-tuning relativamente acessível. Nossos times utilizaram o ajuste LLaMA-Factory's LoRA tuning com sucesso em um modelo LLaMA 7B. Portanto, se você precisa fazer fine-tuning, esta estrutura merece ser avaliada.

96. MLX

Avalie

O **MLX** é um framework de código aberto de arrays, projetado para aprendizado de máquina eficiente e flexível em dispositivos Apple Silicon. Ele permite que cientistas de dados e engenheiros de aprendizado de máquina (ML) acessem a GPU integrada, possibilitando a escolha do hardware mais adequado às suas necessidades. O design do MLX é inspirado por frameworks como NumPy, PyTorch e Jax, entre outros. Um dos principais diferenciais é o modelo de memória unificada do MLX, que elimina a sobrecarga de transferências de dados entre a CPU e a GPU, resultando em execução mais rápida. Esse recurso torna plausível a execução de modelos em dispositivos como iPhones, abrindo uma grande oportunidade para aplicações de IA embarcada. Embora nichado, vale a pena acompanhar o desenvolvimento deste framework para a comunidade de desenvolvedores de ML.

97. Mojo

Avalie

Mojo é uma nova linguagem de programação voltada para IA. Ele visa reduzir a distância entre pesquisa e produção ao combinar a sintaxe e o ecossistema do Python com recursos de programação de sistemas e metaprogramação. É a primeira linguagem a se beneficiar do novo backend de compilação MLIR e traz recursos interessantes como abstração de custo zero, autoajuste, destruição imediata, otimização de chamadas recursivas e melhor ergonomia para SIMD (single instruction, multiple data). Nós gostamos bastante do Mojo e incentivamos você a experimentá-lo. O SDK do Mojo está atualmente disponível para sistemas operacionais Ubuntu e macOS.

98. Otter

Avalie

O **Otter** é uma biblioteca de cache livre de contenção para Go. Embora o Go possua diversas bibliotecas similares, destacamos o Otter por dois motivos: sua excelente taxa de transferência e a implementação inteligente do algoritmo S3-FIFO, que garante uma boa taxa de acerto no cache. Além disso, o Otter oferece suporte a genéricos, permitindo que você utilize qualquer tipo comparável como chave e qualquer tipo como valor.

99. Pkl

Avalie

O **Pkl** é uma linguagem e um conjunto de ferramentas de configuração criados originalmente para uso interno pela Apple e agora disponibilizados como código aberto. A principal característica do Pkl é o seu sistema de tipos e validação, permitindo que erros de configuração sejam detectados antes da implantação. Ele gera arquivos JSON, .plist, YAML e .properties e possui ampla integração com IDEs e outras linguagens, incluindo geração de código.

100. Rust para o desenvolvimento de interface para pessoas usuárias (UI)

Avalie

O impacto do Rust continua crescendo, e muitas das ferramentas de compilação e linha de comando que abordamos recentemente são escritas nele. Agora, também estamos observando uma movimentação no uso de **Rust para o desenvolvimento de interface para pessoas usuárias (UI)**. A maioria das equipes que preferem usar a mesma linguagem para código rodando no navegador e no servidor opta por JavaScript ou TypeScript. No entanto, com o WebAssembly você pode usar Rust no navegador, e isso está se tornando um pouco mais comum agora. Frameworks como Leptos e sauron focam no desenvolvimento web, enquanto Dioxus e vários outros frameworks suportam o desenvolvimento de aplicativos multiplataforma para desktop e dispositivos móveis, além do desenvolvimento web.

101. vLLM

Avalie

O **vLLM** é um motor de inferência e serviço de alta performance e eficiência de memória para modelos de linguagem de grande porte (LLMs). Sua eficiência se deve principalmente à implementação de continuous batching para requisições recebidas. Ele suporta diversas opções de implantação, incluindo inferência e serviço distribuído com paralelismo de tensores usando o runtime Ray, implantação em nuvem com SkyPilot e implantação com NVIDIA Triton, Docker e LangChain. Nossos times tiveram boas experiências executando workers vLLM dockerizados em máquinas virtuais on-prem, integrando com um servidor de API compatível com OpenAI — o qual, por sua vez, é utilizado por diversas aplicações, incluindo plugins de IDE para auxílio à programação e chatbots. Nossas equipes utilizam o vLLM para rodar modelos como CodeLlama 70B, CodeLlama 7B e Mixtral. Outro destaque é a capacidade de escala do motor: bastam algumas alterações na configuração para rodar um modelo de 7B para 70B. Se você busca colocar LLMs em produção, o vLLM merece ser explorado.

102. Voyager

Avalie

O **Voyager** é uma biblioteca de navegação criada para o Jetpack Compose do Android. Ele suporta diversos tipos de navegação, incluindo Linear, BottomSheet, Tab e Nested, e seu modelo de tela integra-se com frameworks populares como Koin e Hilt. Ao usar o Jetpack Compose em um projeto multiplataforma, o Voyager é uma boa opção para implementar um padrão de navegação comum em todas as plataformas suportadas. O desenvolvimento do Voyager voltou a ganhar força e a biblioteca atingiu a versão 1.0 em dezembro de 2023.

103. WGPU

Avalie

wgpu é uma biblioteca gráfica para Rust baseada na API WebGPU, reconhecida por sua capacidade de lidar com tarefas gráficas e de computação de propósito geral na GPU de forma eficiente. O wgpu visa preencher a lacuna deixada pela eliminação gradual de padrões gráficos antigos como OpenGL e WebGL. Ele introduz uma abordagem moderna para o desenvolvimento gráfico que abrange aplicativos nativos e projetos baseados na web. Sua integração com o WebAssembly permite ainda que aplicativos gráficos e de computação sejam executados no navegador. O wgpu representa um avanço para tornar a programação gráfica avançada mais acessível aos desenvolvedores web, com uma variedade de aplicações, desde jogos até a criação de animações web sofisticadas, posicionando o wgpu como uma tecnologia promissora para se avaliar.

104. Zig

Avalie

Zig é uma nova linguagem de programação que compartilha muitos atributos com C, mas com tipagem mais forte, alocação de memória facilitada e suporte a namespaces, entre vários outros recursos. O objetivo do Zig é fornecer uma linguagem simples com compilação direta que minimize efeitos colaterais e entregue execução previsível e fácil de rastrear. O Zig também oferece acesso simplificado à capacidade de compilação cruzada do LLVM. Alguns dos nossos desenvolvedores consideraram esse recurso tão valioso que estão usando o Zig como um compilador cruzado, mesmo não escrevendo código Zig. Vemos equipes na indústria utilizando Zig para auxiliar na construção de cadeias de ferramentas C/C++. Sendo uma linguagem inovadora, o Zig merece ser considerada para aplicações onde C está sendo cogitado ou já esteja em uso.

105. LangChain

Evite

No Radar anterior, mencionamos algumas críticas emergentes sobre o **LangChain**. Desde então, ficamos ainda mais cautelosos em relação a ele. Embora o framework ofereça um conjunto poderoso de recursos para construir aplicações em modelos de linguagem de grande porte (LLMs), consideramos o LangChain difícil de usar e excessivamente complexo. A popularidade e atenção inicial conquistadas na indústria o transformaram em um padrão para muitos. No entanto, à medida que o LangChain tenta evoluir e acompanhar o ritmo acelerado de mudanças, fica cada vez mais difícil para as pessoas desenvolvedoras navegarem por essas alterações em conceitos e padrões. Também descobrimos que o design da API é inconsistente e verboso. Por isso, muitas vezes obscurece o que realmente está acontecendo nos bastidores, dificultando o entendimento e controle do funcionamento dos LLMs e dos diversos padrões ao seu redor. Como consequência, estamos movendo o LangChain para Evite. Em muitos dos nossos casos de uso, verificamos que uma implementação com uso mínimo de frameworks especializados é suficiente. Dependendo do seu caso específico, você também pode considerar outras estruturas como Semantic Kernel, Haystack ou LiteLLM.

Mantenha-se atualizado com os artigos e informações relacionados ao Radar

Assine o Technology Radar para receber emails mensais com insights de tecnologia da Thoughtworks e divulgações das futuras edições.

Assine

A Thoughtworks é uma consultoria global de tecnologia que integra estratégia, design e engenharia de software para alavancar a inovação digital. Somos mais de 10,5 mil pessoas distribuídas entre 48 escritórios e em 19 países. Há mais de 30 anos, trabalhamos junto a nossas clientes para criar impacto extraordinário, usando a tecnologia como diferenciador para ajudá-las a resolver problemas de negócio complexos.

/thoughtworks

Estratégia. Design. Engenharia.