ThoughtWorks®

TECHNOLOGY RADAR

Um guia com opiniões firmes sobre as fronteiras da tecnologia

Volume 23

#TWTechRadar
thoughtworks.com/radar

Contribuições

O Technology Radar é produzido pelo Conselho Consultivo de Tecnologia da ThoughtWorks

O Conselho Consultivo de Tecnologia (TAB) é um grupo formado por 21 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 CTO da ThoughtWorks, Rebecca Parsons.

O TAB atua examinando tópicos que afetam tecnologias e tecnologistas da ThoughtWorks. Com o atual cenário de pandemia global, mais uma vez criamos esta edição do Technology Radar por meio de um evento virtual.


Rebecca Parsons (CTO)


Martin Fowler (Chief Scientist)


Bharani Subramaniam


Birgitta Böckeler


Brandon Byars


Camilla Crispim


Cassie Shum


Erik Dörnenburg


Evan Bottcher


Fausto de la Torre


Hao Xu


Ian Cartwright


James Lewis


Lakshminarasimhan Sudarshan


Mike Mason


Neal Ford


Ni Wang


Perla Villarreal


Rachel Laycock


Scott Shaw


Shangqi Liu


Zhamak Dehghani

Tradução: Alexey Villas Bôas, Angélica de Oliveira, Camilla Crispim, Gregório Melo, Edlaine Zamora, Luiza Souza, Paula Ribas, Patrick Prado, Ricardo Cavalcanti e Tania Gonzales.

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 pessoas interessadas, de pessoas que desenvolvem software a CTOs. O conteúdo é concebido para ser um resumo conciso.

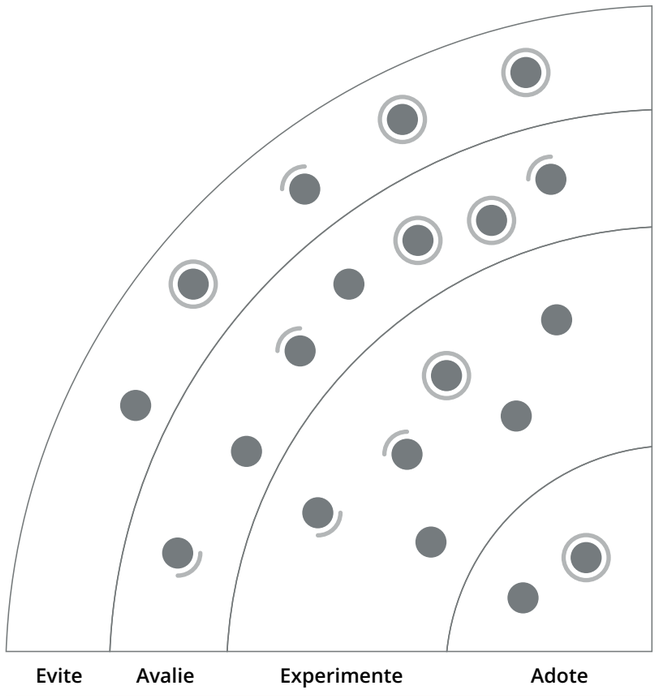
Nós encorajamos você a explorar essas tecnologias. O Radar é gráfico por natureza, agrupando os itens em técnicas, ferramentas, plataformas e linguagens & frameworks. No caso de itens que podem ser classificados em mais de um quadrante, escolhemos aquele que parece mais adequado. Além disso, agrupamos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

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

Radar em um relance

A ideia por trás do Radar é rastrear coisas interessantes, o que chamamos de blips. Organizamos os blips no Radar usando duas categorias: quadrantes e anéis. Os quadrantes representam as diferentes naturezas dos blips. Os anéis indicam em que estágio do ciclo de adoção consideramos que cada blip esteja.

Um blip é uma tecnologia ou técnica que desempenha um papel significativo no desenvolvimento de software. Blips estão sempre em movimento, o que significa que suas posições no Radar estão constantemente mudando — geralmente indicando que nossa confiança neles tem crescido à medida que eles se movimentam entre os anéis.



Novo

Modificado

Sem modificação

Nosso Radar é um olhar para o futuro. Para abrir espaço para novos itens, apagamos itens em que não houve mudança recentemente, o que não é uma representação de seu valor, mas uma solução para nossa limitação de espaço.

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 experimentar esta tecnologia em um projeto que possa lidar com o risco.

Avalie

Vale a pena explorar com o objetivo de compreender como isso afetará sua empresa.

Evite

Prossiga com cautela.

Temas desta edição

A imponência do GraphQL

Vemos um aumento na adoção do GraphQL em muitos times, acompanhado por um ecossistema de suporte também em crescimento. Ele resolve alguns problemas comuns que se manifestam em arquiteturas distribuídas modernas, como microsserviços: quando as pessoas desenvolvedoras dividem as coisas em pequenos pedaços, elas devem reagregar informações com frequência para resolver os requisitos de negócio.
O GraphQL oferece recursos convenientes para resolver esse problema cada vez mais comum. Como todas as abstrações poderosas, ele oferece compensações e requer consideração cuidadosa por parte dos times para evitar efeitos negativos no longo prazo. Por exemplo, vemos times fornecendo muitas informações sobre os detalhes de implementação subjacentes por meio de uma ferramenta de agregação, levando a uma fragilidade desnecessária na arquitetura. Outro benefício de curto prazo que pode se transformar em dor de cabeça a longo prazo surge quando os times tentam usar uma ferramenta de agregação para criar um modelo de dados canônico, universal e centralizado. Encorajamos os times a usar GraphQL e as ferramentas emergentes ao seu redor, mas tendo cautela com as tecnologias de foco específico que são generalizadas para resolver muitos problemas.

A luta com o navegador continua

O navegador web foi originalmente projetado para navegação e interação com documentos, mas hoje ele essencialmente hospeda aplicações, e a incompatibilidade de abstrações continua desafiando as pessoas desenvolvedoras. Para superar as muitas dores de cabeça inerentes a essa incompatibilidade, os times de desenvolvimento continuam repensando e desafiando as abordagens estabelecidas para testes de navegador, gerenciamento de estado e construção de aplicações ricas e rápidas para navegadores de forma geral. Vemos várias dessas tendências no Radar. Em primeiro lugar, desde que movemos o Redux para o anel Adote em 2017, como forma padrão de gerenciar o estado em aplicações React, vemos agora pessoas desenvolvedoras procurando em outros lugares (Recoil), ou adiando a decisão por uma biblioteca de gerenciamento de estado. Em segundo lugar, Svelte vem conquistando mais interesse e desafiando um dos conceitos estabelecidos e aplicados por frameworks de aplicações populares como React e Vue.js: o virtual DOM. Em terceiro lugar, continuamos vendo novas ferramentas para lidar com testes no navegador: Playwright é mais uma tentativa de melhorar os testes de UI, e Mock Service Worker é uma nova abordagem para desacoplar testes de suas interações back-end. Em quarto lugar, continuamos observando o desafio de equilibrar a produtividade de pessoas desenvolvedoras com desempenho, com os polyfills adaptados ao navegador buscando mover a escala nessa balança.

Visualize tudo

Este volume do Radar traz diversos blips sobre variadas áreas da tecnologia, mas com uma coisa em comum: visualização. Você encontrará blips sobre infraestrutura, ciência de dados, recursos de nuvem e uma série de outras ferramentas de visualização inovadoras, incluindo algumas formas muito eficazes de visualizar abstrações difíceis. Você também encontrará discussões sobre ferramentas de visualização de dados interativas e ferramentas de painel, como Dash, Bokeh e Streamlit, bem como uma série de ferramentas de visualização de infraestrutura, incluindo Kiali para visualização de malhas de serviço em arquiteturas de microsserviços. À medida que os ecossistemas de desenvolvimento se tornam mais complexos, uma imagem geralmente ajuda muito a gerenciar a inevitável sobrecarga cognitiva.

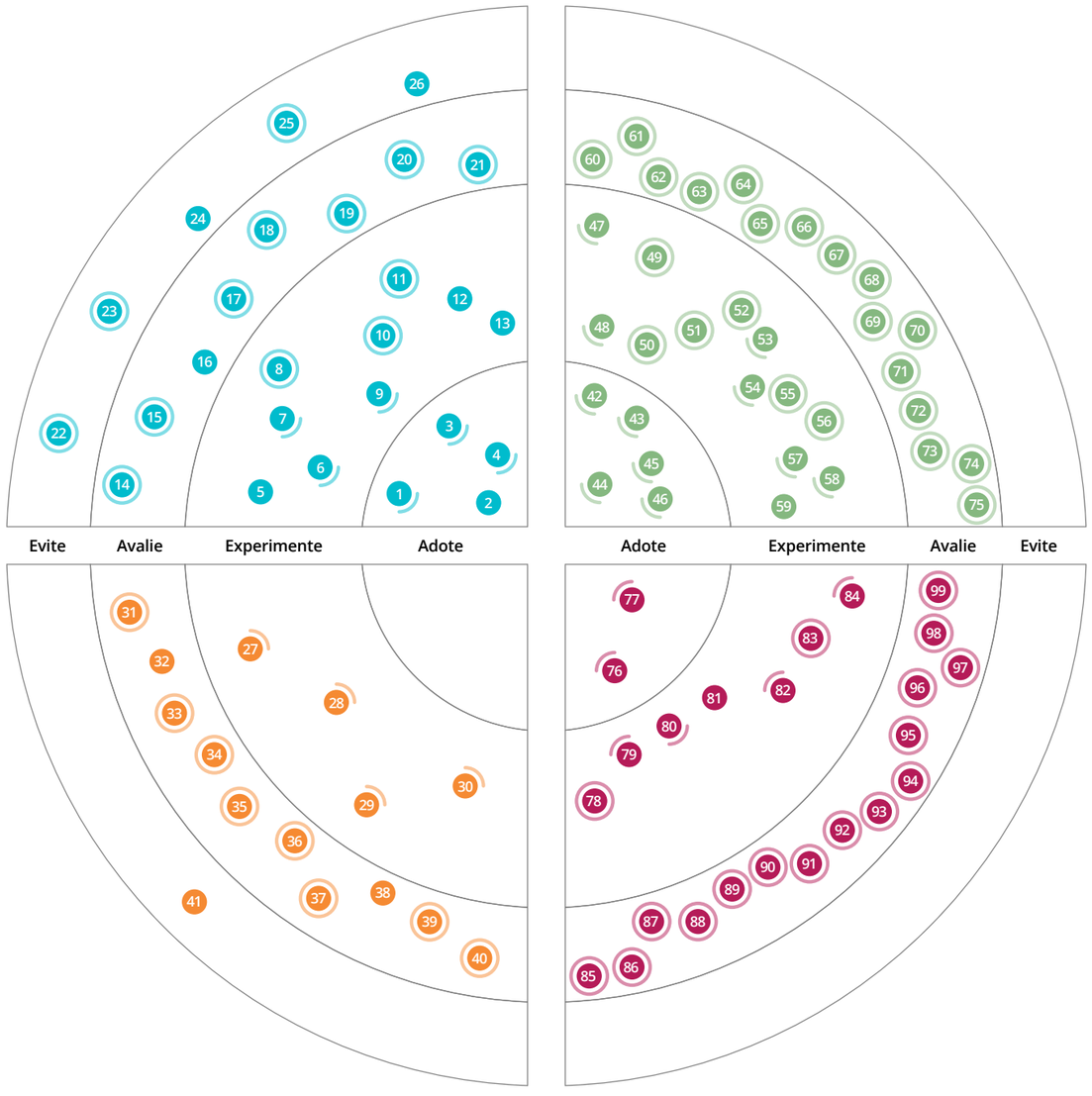
A adolescência da infraestrutura como código

Gerenciar infraestrutura como código tornou-se mais comum à medida que as organizações enxergam os benefícios de automatizar a infraestrutura. Consequentemente, cria-se um ciclo de feedback de adoção e inovação para quem cria ferramentas e frameworks. Ferramentas como CDK e Pulumi, entre outras, oferecem recursos que vão muito além da primeira geração, evoluindo de tal forma que acreditamos que a infraestrutura como código chegou à adolescência — com todas as suas conotações positivas e negativas. Foi uma surpresa agradável o número de blips, em todos os quadrantes, refletindo positivamente o aumento da maturidade do ecossistema. No entanto, também discutimos os desafios em torno da falta de padrões maduros e as dificuldades que muitas empresas enfrentam ao tentar encontrar o melhor uso desse recurso — o que indica crescimento contínuo em direção à maturidade. Temos esperança de que a comunidade de infraestrutura continuará aprendendo lições de design de software, especialmente em termos de criação de uma infraestrutura implantável com fraco acoplamento.

Democratizando a programação

Várias de nossas discussões giram em torno de ferramentas e técnicas que promovem a democratização da programação: ou seja, habilitar as pessoas que não programam a realizar tarefas que até então apenas as pessoas programadoras poderiam fazer.
Soluções como IFTTT e Zapier, por exemplo, são populares neste espaço há bastante tempo. Temos observado um uso crescente de ferramentas como Amazon Honeycode, um ambiente de baixo código para a criação de aplicações de negócio simples. Embora ferramentas como essas forneçam ambientes de programação adequados à finalidade, os desafios surgem ao movê-los para ambientes de produção em escala. Pessoas desenvolvedoras e especialistas em planilhas há muito tempo conseguem encontrar um meio-termo entre ambientes de programação específicos de domínio e tradicionais. O advento de ferramentas mais modernas renova essa discussão em domínios mais amplos, com muitos dos mesmos desdobramentos positivos e negativos.

O Radar



Técnicas

Adote
1. Função de aptidão para controle de dependências
2. Custo de execução como função de aptidão arquitetural
3. Políticas de segurança como código
4. Modelos de serviço personalizados

Experimente
5. Entrega contínua para aprendizado de máquina (CD4ML)
6. Malha de dados
7. Definição de pipeline de dados declarativa
8. Diagramas como código
9. Imagens Docker sem distribuição
10. Interceptação de eventos
11. Execução paralela com reconciliação
12. Use abordagens e processos "nativamente remotos"
13. Arquitetura de confiança zero

Avalie
14. Plataformas limitadas de baixo código
15. Polyfills adaptados ao navegador
16. Identidade descentralizada
17. Serviços em nuvem gerenciados por Kube
18. Open Application Model (OAM)
19. Enclaves seguros
20. Experimentação de switchback
21. Credenciais verificáveis

Evite
22. Apollo Federation
23. ESBs disfarçados de gateways de API
24. Agregação de logs para análise de negócio
25. Anarquia de micro frontends
26. Uso de notebooks em produção

Plataformas

Adote

Experimente
27. Azure DevOps
28. Debezium
29. Honeycomb
30. JupyterLab

Avalie
31. Amundsen
32. AWS Cloud Development Kit
33. Backstage
34. Dremio
35. DuckDB
36. K3s
37. Materialize
38. Pulumi
39. Tekton
40. Trust over IP stack

Evite
41. Uso excessivo de Node.js

Ferramentas

Adote
42. Airflow
43. Bitrise
44. Dependabot
45. Helm
46. Trivy

Experimente
47. Bokeh
48. Concourse
49. Dash
50. jscodeshift
51. Kustomize
52. MLflow
53. Pitest
54. Sentry
55. ShellCheck
56. Stryker
57. Terragrunt
58. tfsec
59. Yarn

Avalie
60. CML
61. Eleventy
62. Flagger
63. gossm
64. Great Expectations
65. k6
66. Katran
67. Kiali
68. LGTM
69. Litmus
70. Opacus
71. OSS Index
72. Playwright
73. pnpm
74. Sensei
75. Zola

Evite

Linguagens & Frameworks

Adote
76. Arrow
77. jest-when

Experimente
78. Fastify
79. Immer
80. Redux
81. Rust
82. single-spa
83. Strikt
84. XState

Avalie
85. Babylon.js
86. Blazor
87. Flutter Driver
88. HashiCorp Sentinel
89. Hermes
90. io-ts
91. Kedro
92. LitElement
93. Mock Service Worker
94. Recoil
95. Snorkel
96. Streamlit
97. Svelte
98. SWR
99. Testing Library

Evite

TECHNOLOGY RADAR

Técnicas

Técnicas

Função de aptidão para controle de dependências

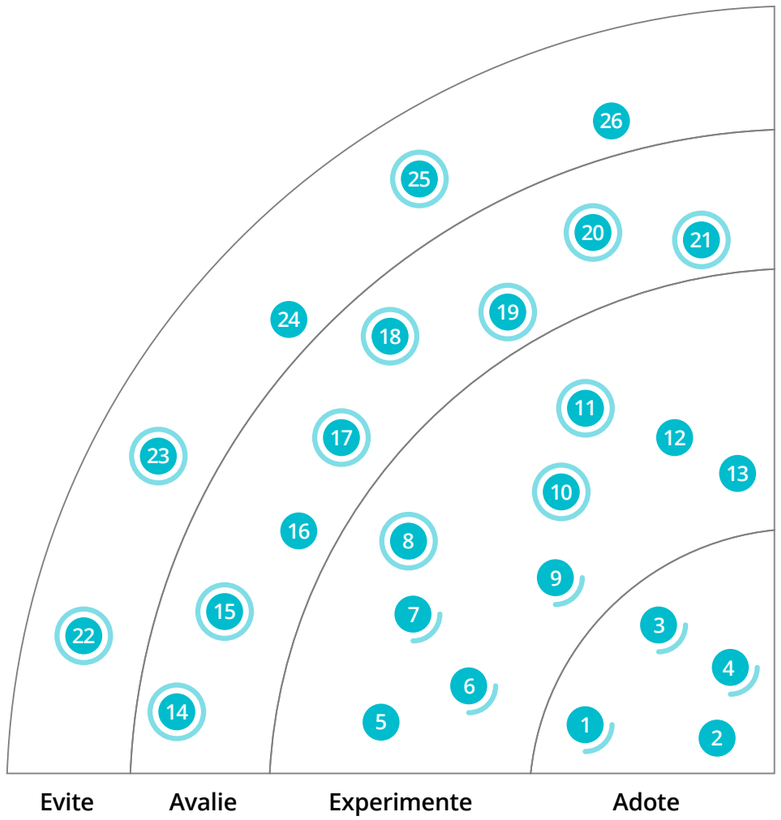
Adote

As função de aptidão introduzidas pela arquitetura evolutiva e emprestadas pela computação evolucionária, são funções executáveis que nos informam se nossas aplicações e arquitetura estão objetivamente se afastando de suas características desejadas. São essencialmente testes que podem ser incorporados em nossos pipelines de entrega de software. Uma das principais características de uma aplicação é a atualização de suas dependências para outras bibliotecas, APIs ou componentes de um ambiente. Uma função de aptidão para controle de dependências rastreia esses elementos para sinalizar as dependências desatualizadas que requerem atualização. Com o número crescente de ferramentas para controle de dependências, como Dependabot ou Snyk, podemos facilmente incorporar funções de aptidão para controle de dependências em nosso processo de entrega de software, habilitando a adoção de medidas oportunas para manter as dependências de nossas aplicações atualizadas.

Custo de execução como função de aptidão arquitetural

Adote

Automatizar estimativa, rastreamento e projeção do custo de execução de uma infraestrutura de nuvem é necessário para as empresas de hoje. Os modelos de precificação sagazes dos fornecedores de nuvem, combinados com a proliferação dos parâmetros de precificação e a natureza dinâmica da arquitetura atual podem levar a um custo de execução surpreendentemente caro. Por exemplo, os preços de arquiteturas sem servidor baseadas em chamadas de API, soluções de streaming de eventos baseadas em tráfego ou clusters de processamento de dados baseados em trabalhos em execução, todos têm uma natureza dinâmica que muda com o tempo à medida que a arquitetura evolui. Quando nossos times gerenciam infraestruturas na nuvem, implementar custo de execução como função de aptidão arquitetural é uma das primeiras atividades. Isso significa que nossos times podem observar o custo de executar serviços em relação ao valor entregue. Quando observam divergências em relação ao que era esperado ou aceitável, discutem se é hora de evoluir a arquitetura. A observação e o cálculo do custo de execução são implementados como uma função automatizada.



Políticas de segurança como código

Adote

À medida que o panorama da tecnologia se torna mais complexo, questões como segurança demandam mais automação e práticas de engenharia.

Adote
1. Função de aptidão para controle de dependências
2. Custo de execução como função de aptidão arquitetural
3. Políticas de segurança como código
4. Modelos de serviço personalizados

Experimente
5. Entrega contínua para aprendizado de máquina (CD4ML)
6. Malha de dados
7. Definição de pipeline de dados declarativa
8. Diagramas como código
9. Imagens Docker sem distribuição
10. Interceptação de eventos
11. Execução paralela com reconciliação
12. Use abordagens e processos "nativamente remotos"
13. Arquitetura de confiança zero

Avalie
14. Plataformas limitadas de baixo código
15. Polyfills adaptados ao navegador
16. Identidade descentralizada
17. Serviços em nuvem gerenciados por Kube
18. Open Application Model (OAM)
19. Enclaves seguros
20. Experimentação de switchback
21. Credenciais verificáveis

Evite
22. Apollo Federation
23. ESBs disfarçados de gateways de API
24. Agregação de logs para análise de negócio
25. Anarquia de micro frontends
26. Uso de notebooks em produção

Técnicas

A malha de dados marca uma mudança de paradigma na forma como gerenciamos grandes dados analíticos, em uma tentativa de abordar muitos dos desafios conhecidos do gerenciamento centralizado de dados analíticos anterior. Há, entretanto, uma escassez de ferramentas comerciais ou de código aberto para dar suporte à malha de dados.

(Malha de dados)

Ao construir sistemas, precisamos levar em consideração as políticas de segurança, que consistem em regras e procedimentos para proteger nossos sistemas contra ameaças e disrupções. Por exemplo, as políticas de controle de acesso definem e impõem quem pode acessar quais serviços e recursos em quais circunstâncias. Por outro lado, as políticas de segurança de rede podem limitar dinamicamente a taxa de tráfego para um serviço específico.

Vários de nossos times tiveram boas experiências tratando políticas de segurança como código. Quando dizemos como código, não significa apenas escrever essas políticas de segurança em um arquivo, mas também aplicar práticas como manter o código sob controle de versão, introduzir validação automática no pipeline, implantá-lo automaticamente nos ambientes e observar e monitorar seu desempenho. Com base em nossa experiência e maturidade das ferramentas existentes — incluindo Open Policy Agent e plataformas como Istio, que fornece definição de política flexível e mecanismos de aplicação que suportam a prática de política de segurança como código —, é altamente recomendável usar essa técnica em seu ambiente.

Modelos de serviço personalizados

Adote

Desde que mencionamos pela última vez os modelos de serviço personalizados, vimos uma adoção mais ampla do padrão ajudando a pavimentar o caminho de organizações que estão migrando para microsserviços. Com avanços constantes em ferramentas de observabilidade, orquestração de contêineres e sidecars de malha de serviço, os modelos fornecem padrões razoáveis para iniciar novos serviços, dispensando uma grande quantidade de configurações necessárias para garantir que o serviço funcione bem com a infraestrutura ao seu redor. Tivemos sucesso aplicando princípios de gestão de produto a modelos de serviço personalizados, tratando pessoas desenvolvedoras internas como clientes e facilitando o processo de enviar código para produção e operá-lo com a observabilidade apropriada. Essa prática traz o benefício adicional de atuar como um mecanismo de governança leve para centralizar as decisões técnicas padrão.

Entrega contínua para aprendizado de máquina (CD4ML)

Experimente

Cerca de uma década atrás, apresentamos a entrega contínua (CD), nossa maneira padrão de fornecer soluções de software. As soluções de hoje incluem cada vez mais modelos de aprendizado de máquina e não os consideramos exceção para adoção de práticas de entrega contínua. Chamamos isso de entrega contínua para aprendizado de máquina (CD4ML). Embora os princípios de CD permaneçam os mesmos, as práticas e ferramentas para implementar o processo de treinamento, teste, implantação e monitoramento de modelos de ponta a ponta demandam algumas modificações. Por exemplo: o controle de versão não deve incluir apenas o código, mas também os dados, os modelos e seus parâmetros; a pirâmide de testes se estende para incluir a validação de vieses, imparcialidade e dados e recursos do modelo; o processo de implantação deve considerar como promover e avaliar o desempenho de novos modelos em relação aos modelos de referência atuais. Embora o setor esteja entusiasmado com a nova palavra da moda, MLOps, sentimos que CD4ML é nossa abordagem holística para implementação de um processo de ponta a ponta para liberar de forma confiável e melhorar continuamente os modelos de aprendizado de máquina, da ideia à produção.

Malha de dados

Experimente

A malha de dados marca uma mudança bem-vinda de paradigma arquitetural e organizacional na forma como gerenciamos grandes dados analíticos. O paradigma é baseado em quatro princípios: (1) descentralização orientada a domínio da propriedade e da arquitetura de dados; (2) dados orientados a domínio e servidos como produto; (3) infraestrutura de dados de autoatendimento como uma plataforma para habilitar times autônomos de dados orientados a domínio; (4) governança federada para habilitar ecossistemas e interoperabilidade. Embora os princípios sejam intuitivos e tentem abordar muitos dos desafios conhecidos do gerenciamento centralizado de dados analíticos anterior, eles transcendem as tecnologias de dados analíticos disponíveis. Depois de construir a malha de dados para uma variedade de clientes em cima de ferramentas existentes, aprendemos duas coisas: (a) faltam ferramentas de código aberto ou comerciais para a aceleração da implementação da malha de dados (por exemplo, uma ferramenta para implementação de um modelo de acesso universal para dados poliglotas baseados em tempo, o que atualmente criamos de forma personalizada para clientes) e (b) apesar da indisponibilidade de ferramentas, é possível usar tecnologias existentes como blocos de construção básicos.

Naturalmente, a adaptação da tecnologia é um componente importante da implementação de uma estratégia de dados com base na malha de dados em sua organização. O sucesso, no entanto, exige uma reestruturação organizacional para separar o time de plataforma de dados, a criação da função de product owner de dados para cada domínio e a introdução das estruturas de incentivo necessárias para que os domínios tenham propriedade e compartilhem seus dados analíticos como produtos.

Definição de pipeline de dados declarativa

Experimente

Muitos pipelines de dados são definidos em um script grande, mais ou menos imperativo, escrito em Python ou Scala. O script contém a lógica das etapas individuais, bem como o código que as une. Quando confrontadas com uma situação semelhante nos testes Selenium, as pessoas desenvolvedoras descobriram o padrão Page Object e, posteriormente, muitos frameworks de desenvolvimento orientados a comportamento (BDD) implementaram uma divisão entre as definições de etapas e sua composição. Agora, alguns times estão tentando trazer o mesmo pensamento para a engenharia de dados. Uma definição de pipeline de dados declarativa separada, talvez escrita em YAML, contém apenas a declaração e a sequência de etapas. Ela indica conjuntos de dados de entrada e saída, mas faz referência a scripts se e quando uma lógica mais complexa for necessária. A La Mode é uma ferramenta relativamente nova que usa uma abordagem DSL para definir pipelines, mas airflow-declarative, uma ferramenta que transforma grafos acíclicos direcionados definidos em YAML em agendas de tarefas Airflow, parece ter mais força neste espaço.

Diagramas como código

Experimente

Temos visto cada vez mais ferramentas que permitem criar arquitetura de software e outros diagramas como código. Há benefícios em usar essas ferramentas em relação às alternativas mais pesadas, incluindo controle de versão simples e capacidade de gerar DSLs de várias fontes. Entre as ferramentas de que gostamos neste espaço estão Diagrams, Structurizr DSL, AsciiDoctor Diagram e estáveis como WebSequenceDiagrams, PlantUML e o respeitável Graphviz. Também é bastante simples gerar seu próprio SVG atualmente, portanto, também não descarte a possibilidade de criar rapidamente sua própria ferramenta. Um de nossos autores escreveu um pequeno script Ruby para criar SVGs rapidamente, por exemplo.

Imagens Docker sem distribuição

Experimente

Ao criar imagens do Docker para nossas aplicações, geralmente temos duas preocupações: a segurança e o tamanho da imagem. Tradicionalmente, usamos ferramentas de escaneamento de segurança de contêiner para detectar e corrigir vulnerabilidades e exposições comuns, e pequenas distribuições como Alpine Linux para tratar do tamanho da imagem e do desempenho da distribuição. Agora, temos mais experiência com imagens do Docker sem distribuição e podemos recomendar essa abordagem como mais uma precaução de segurança importante para aplicações em contêineres. Imagens do Docker sem distribuição reduzem o rastro e as dependências ao eliminar uma distribuição completa do sistema operacional. Essa técnica reduz o ruído da verificação de segurança e a superfície de ataque da aplicação. São menos vulnerabilidades a serem corrigidas e, como bônus, essas imagens menores são mais eficientes. O Google publicou um conjunto de imagens de contêiner sem distribuição para diferentes linguagens. Você pode criar imagens de aplicações sem distribuição usando a ferramenta de construção do Google Bazel ou simplesmente usar Dockerfiles de vários estágios (multistage). Observe que os contêineres sem distribuição, por padrão, não têm um shell para depuração. No entanto, você pode encontrar facilmente versões de depuração de contêineres sem distribuição online, incluindo um shell BusyBox. O uso de imagens do Docker sem distribuição é uma técnica pioneira do Google e, em nossa experiência, ainda é muito restrita a imagens geradas pela empresa. Esperamos que a técnica avance para além deste ecossistema.

Interceptação de eventos

Experimente

À medida que mais empresas substituem seus sistemas legados, achamos que vale a pena destacar uma alternativa para a captura de dados alternados (CDC) como um mecanismo para obter dados desses sistemas. Martin Fowler descreveu a interceptação de eventos em 2004. Em termos modernos, ela envolve a bifurcação de solicitações na entrada em um sistema para que seja possível

Técnicas

Estamos observando uma proliferação de ferramentas que permitem criar arquitetura de software e outros diagramas como código. Os benefícios incluem controle de versão simples e capacidade de gerar DSLs de várias fontes.

(Diagramas como código)

Técnicas

A interceptação de eventos é uma alternativa válida para alterar a captura de dados ao substituir sistemas legados. Obter mudanças de estado da fonte, em vez de tentar recriá-las pós-processamento usando CDC, tem sido uma técnica negligenciada.

(Interceptação de eventos)

gradualmente construir um substituto. Frequentemente, isso é feito copiando eventos ou mensagens, mas a bifurcação de solicitações HTTP é igualmente válida. Os exemplos incluem bifurcação de eventos de sistemas de ponto de venda antes de serem gravados em um mainframe e bifurcação de transações de pagamento antes de serem gravadas em um sistema bancário central. Ambos levam à substituição gradual de partes dos sistemas legados. Sentimos que, como técnica, a obtenção de mudanças de estado da fonte, ao invés de tentar recriá-las pós-processamento usando CDC, tem sido esquecida, por isso estamos dando destaque a ela nesta edição do Radar.

Execução paralela com reconciliação

Experimente

Substituir código legado em escala é sempre uma tarefa difícil, que geralmente se beneficia da execução paralela com reconciliação. Na prática, a técnica se baseia na execução do mesmo fluxo de produção por meio do código antigo e do novo, retornando a resposta do código legado, mas comparando os resultados para ganhar confiança no novo código. Apesar de ser uma técnica antiga, vimos implementações mais robustas nos últimos anos, tendo como base práticas de entrega contínua, como implantações canário e feature toggles, e estendendo-as com a adição de uma camada extra de experimentação e análise de dados para comparar os resultados em tempo real. Usamos a abordagem até mesmo para comparar resultados interfuncionais, como tempo de resposta. Embora tenhamos usado a técnica várias vezes com ferramentas feitas sob medida, certamente devemos um reconhecimento à ferramenta Scientist, do GitHub, usada para modernizar uma parte crítica da aplicação e que agora foi disponibilizada para várias linguagens.

Use abordagens e processos "nativamente remotos"

Experimente

Conforme a pandemia se estende, parece que os times altamente distribuídos serão o "novo normal", pelo menos por enquanto. Nos últimos seis meses, aprendemos muito sobre trabalho remoto eficaz. Como pontos positivos, boas ferramentas visuais para gerenciamento de trabalho e colaboração tornaram mais fácil do que nunca colaborar remotamente com colegas. Os times de desenvolvimento, por exemplo, podem contar com o Visual Studio Live Share e o GitHub Codespaces para facilitar o trabalho coletivo e aumentar a produtividade. A maior desvantagem do trabalho remoto pode ser o esgotamento: muitas pessoas têm as agendas tomadas por videochamadas em sequência durante todo o dia, e essa dinâmica começa a ter consequências. Embora as ferramentas visuais online facilitem a colaboração, também é possível construir diagramas grandes e complexos que acabam sendo muito difíceis de usar, e os aspectos de segurança da proliferação de ferramentas também precisam ser gerenciados com cuidado. Nosso conselho é lembrar de dar um passo para trás, conversar com seus times, avaliar o que está funcionando e o que não está e alterar processos e ferramentas conforme necessário.

Arquitetura de confiança zero

Experimente

Enquanto a estrutura de computação e dados continua a mudar nas empresas — de aplicações monolíticas para microsserviços, de lagos de dados centralizados para malhas de dados, de hospedagem local a policloud, com uma crescente proliferação de dispositivos conectados — a abordagem para proteger ativos empresariais em sua maior parte permanece inalterada, com grande dependência e confiança no perímetro da rede: as organizações continuam fazendo investimentos pesados para proteger seus ativos, fortalecendo os muros virtuais de suas empresas, usando configurações de firewall e links privados, e substituindo processos de segurança estáticos e complexos que não atendem mais à realidade de hoje. Essa tendência contínua nos obrigou a destacar a arquitetura de confiança zero (ZTA) novamente.

A ZTA é uma mudança de paradigma na arquitetura e na estratégia de segurança. Parte-se do pressuposto de que um perímetro de rede não representa mais um limite seguro, e nenhuma confiança implícita deve ser concedida a usuários ou serviços com base exclusivamente em sua localização física ou de rede. O número de recursos, ferramentas e plataformas disponíveis para implementar

aspectos da ZTA continua crescendo, e inclui: aplicação de políticas como código, com base no menor privilégio e princípios mais granulares possíveis, além de monitoramento contínuo e mitigação automatizada de ameaças; uso da malha de serviços para impor o controle de segurança aplicação a serviço e serviço a serviço; implementação de atestado binário para verificação da origem dos binários; e inclusão de enclaves seguros, além da criptografia tradicional, para reforçar os três pilares da segurança de dados: em trânsito, em repouso e na memória. Para obter uma introdução ao tópico, consulte a publicação NIST ZTA e o artigo do Google sobre BeyondProd.

Plataformas limitadas de baixo código

Avalie

Uma das decisões mais complexas que as empresas enfrentam no momento é a adoção de plataformas de baixo código ou sem código, ou seja, plataformas que resolvem problemas muito específicos em domínios muito limitados. Muitas fornecedoras estão ocupando agressivamente este espaço. Os problemas que vemos com essas plataformas normalmente estão relacionados à incapacidade de aplicar boas práticas de engenharia, como controle de versão. Testar também é normalmente muito difícil. No entanto, notamos algumas novas participantes interessantes no mercado — incluindo Amazon Honeycode, que facilita a criação de aplicações simples de gerenciamento de tarefas ou eventos, e Parabola, para fluxos de trabalho em nuvem do tipo IFTTT. Por esse motivo, estamos incluindo as plataformas limitadas de baixo código nesta edição. No entanto, continuamos com dúvidas sobre sua aplicabilidade mais ampla, uma vez que essas ferramentas, assim como algumas espécies de plantas, têm por característica avançar para além de seus limites e se emaranhar com outras. Por isso, ainda recomendamos cautela em sua adoção.

Polyfills adaptados ao navegador

Avalie

Polyfills são extremamente úteis para ajudar a evoluir a web, fornecendo implementações que substituem recursos modernos para navegadores que não os implementam (ainda). Muitas vezes, porém, as aplicações web enviam polyfills para navegadores que não precisam deles, o que gera downloads desnecessários e sobrecarga de análise sintática. A situação está se tornando mais evidente agora, já que apenas alguns mecanismos de renderização permanecem em uso e a maior parte dos polyfills visa apenas um deles: o renderizador Trident no IE11. Além disso, a participação de mercado do IE11 está diminuindo com o encerramento do suporte em menos de um ano. Portanto, sugerimos que você use polyfills adaptados ao navegador, enviando apenas os polyfills necessários para um determinado navegador. Essa técnica pode até mesmo ser implementada como serviço com o Polyfill.io.

Identidade descentralizada

Avalie

Em 2016, Christopher Allen, um importante contribuidor no espaço de SSL/TLS, nos inspirou com a introdução de 10 princípios sustentando uma nova forma de identidade digital e um caminho para chegar lá, o caminho para a identidade auto-soberana. A identidade auto-soberana, também conhecida como identidade descentralizada, é uma "identidade portátil vitalícia para qualquer pessoa, organização ou coisa que não dependa de nenhuma autoridade centralizada e jamais possa ser removida", de acordo com o padrão Trust over IP. A adoção e a implementação da identidade descentralizada vêm ganhando impulso e se tornando algo mais possível de alcançar. Vemos sua adoção em aplicações de saúde para clientes, infraestruturas de saúde governamentais e identidades jurídicas corporativas que respeitam a privacidade. Se você deseja adotar rapidamente a identidade descentralizada, pode avaliar sistemas de suporte como Sovrin Network, Hyperledger Aries e Indy, bem como padrões para identificadores descentralizados e credenciais verificáveis. Estamos observando este espaço de perto, enquanto ajudamos nossa base de clientes com seus posicionamentos estratégicos na nova era de confiança digital.

Serviços em nuvem gerenciados por Kube

Avalie

As provedoras de nuvem começaram lentamente a oferecer suporte a APIs estilo Kubernetes, por meio de definições de recursos personalizadas (CRDs), para gerenciar seus serviços de nuvem. Na maioria dos casos, esses serviços em nuvem são uma parte central da infraestrutura, e observamos times usarem ferramentas como Terraform ou Pulumi para provisioná-los.
Com os novos CRDs (ACK para AWS, Azure Service Operator para Azure e Config Connectors para GCP), você pode usar Kubernetes para provisionar e gerenciar esses serviços em nuvem. Uma vantagem dos serviços de nuvem gerenciados por Kube é que você pode aproveitar o mesmo plano de controle

Técnicas

Plataformas de baixo código ou sem código podem resolver problemas muito específicos em domínios muito limitados. No entanto, mantemos nosso ceticismo sobre sua aplicabilidade de forma mais ampla.

(Plataformas limitadas de baixo código)

Técnicas

A maioria das credenciais digitais hoje são registros de dados simples de sistemas de informação, fáceis de modificar e falsificar, e frequentemente expõem informações desnecessárias. Nos últimos anos, vimos a maturidade crescente das credenciais verificáveis resolver esse problema.

(Credenciais verificáveis)

do Kubernetes para forçar o estado declarativo tanto da sua aplicação quanto da sua infraestrutura. A desvantagem é que ele acopla fortemente o cluster do Kubernetes com a infraestrutura. Portanto, estamos avaliando-o cuidadosamente, e você também deve ter cautela.

Open Application Model (OAM)

Avalie

Falamos muito sobre os benefícios da criação de times de produto de engenharia de plataforma em apoio aos outros times de produto, mas colocar isso em prática é difícil. A indústria parece ainda estar procurando a abstração certa no universo da infraestrutura como código. Embora ferramentas como Terraform e Helm sejam passos na direção certa, o foco ainda está no gerenciamento da infraestrutura, em oposição ao desenvolvimento de aplicações. Também há mudanças em relação ao conceito de infraestrutura como software, com novas ferramentas como Pulumi e CDK sendo lançadas. O Open Application Model (OAM) é uma tentativa de trazer alguma padronização para este espaço. Usando abstrações de componentes, configurações de aplicativos, escopos e características, as pessoas desenvolvedoras podem descrever suas aplicações de uma forma agnóstica de plataforma, enquanto as implementadoras da plataforma podem defini-la em termos de carga de trabalho, característica e escopo. Resta saber se o OAM será amplamente adotado, mas recomendamos ficar de olho nesta ideia interessante e necessária.

Enclaves seguros

Avalie

Enclaves seguros, também identificados como ambientes de execução confiáveis (TEE), referem-se a uma técnica que isola um ambiente — processador, memória e armazenamento — com um nível mais alto de segurança, fornecendo somente uma troca limitada de informações com seu contexto de execução circundante não-confiável. Por exemplo, um enclave seguro nos níveis de hardware e sistema operacional pode criar e armazenar chaves privadas e executar operações com elas, como criptografar dados ou verificar assinaturas, sem que as chaves privadas saiam do enclave seguro ou sejam carregadas na memória da aplicação não-confiável. O enclave seguro fornece um conjunto limitado de instruções para executar operações confiáveis, isoladas de um contexto de aplicação não-confiável.

A técnica é há bastante tempo suportada por muitas fornecedoras de hardware e sistemas operacionais (incluindo a Apple), e as pessoas desenvolvedoras vêm a usando em aplicações de IoT e edge. Porém, apenas recentemente ela ganhou atenção em aplicações corporativas e baseadas em nuvem. As provedoras de nuvem começaram a introduzir recursos de computação confidencial, como enclaves seguros baseados em hardware: a infraestrutura de computação confidencial da Azure promete VMs habilitadas para TEE e acesso por meio da biblioteca de código aberto Open Enclave SDK para realizar operações confiáveis. Da mesma forma, o VMs confidenciais e Compute Engine, do GCP, ainda em beta, possibilita o uso de VMs com criptografia de dados na memória, e o AWS Nitro Enclaves está seguindo os mesmos passos com seu próximo lançamento de visualização. Com a introdução de enclaves seguros baseados em nuvem e computação confidencial, podemos adicionar um terceiro pilar à proteção de dados: em repouso, em trânsito e, agora, na memória.

Embora ainda estejamos no início da jornada de enclaves seguros para empresas, encorajamos você a considerar essa técnica e se manter em dia com as informações sobre vulnerabilidades conhecidas que podem comprometer os enclaves seguros das fornecedoras de hardware subjacentes.

Experimentação de switchback

Avalie

Experimentos controlados usando testes A/B são uma ótima maneira de embasar as decisões sobre o desenvolvimento de produtos. Mas não funcionam bem quando não podemos estabelecer independência entre os dois grupos envolvidos no teste A/B — ou seja, adicionar alguém ao grupo “A” afeta o grupo “B” e vice-versa. Uma técnica para resolver essa categoria de problema é a experimentação de switchback. O conceito central aqui é a alternância entre os modos “A” e “B” do experimento em uma determinada região, em períodos alternados, em vez de ambos executando durante o mesmo período. Em seguida, comparamos a experiência de cliente e outras métricas importantes entre os dois intervalos de tempo. Nós testamos essa técnica com bons resultados em alguns de nossos projetos — é uma boa ferramenta para ter em nossa caixa de ferramentas para experimentos.

Credenciais verificáveis

Avalie

As credenciais estão por toda parte em nossas vidas, incluindo passaportes, carteiras de habilitação e certificados acadêmicos. No entanto, a maioria das credenciais digitais hoje são registros de dados simples de sistemas de informação, fáceis de modificar e falsificar, e frequentemente expõem informações desnecessárias. Nos últimos anos, vimos o amadurecimento contínuo das credenciais verificáveis resolver esse problema. O padrão W3C define de uma forma criptograficamente segura, que respeita

a privacidade e é verificável por máquina. O modelo coloca titulares de credenciais no centro, o que é semelhante à nossa experiência ao usar credenciais físicas: as pessoas usuárias podem colocar suas credenciais verificáveis em suas próprias carteiras digitais e mostrá-las a qualquer momento, sem necessidade de permissão da entidade emissora das credenciais. Essa abordagem descentralizada também permite que as pessoas usuárias gerenciem melhor suas próprias informações, podendo selecionar o que compartilhar e melhorando significativamente a proteção da privacidade dos dados. Por exemplo, usando tecnologia de prova de conhecimento zero, você pode criar uma credencial verificável para provar que é uma pessoa adulta sem revelar sua data de nascimento. A comunidade desenvolveu muitos casos de uso em torno das credenciais verificáveis. Nós implementamos nossa própria certificação para a COVID, tendo como referência a COVID-19 Credentials Initiative (CCI). Embora as credenciais verificáveis não dependam de tecnologia de blockchain ou de identidade descentralizada, essa técnica frequentemente funciona com identidade descentralizada na prática e usa blockchain como registro de dados verificáveis. Muitos frameworks de identidade descentralizada também incorporam credenciais verificáveis.

Apollo Federation

Evite

Quando abordamos GraphQL pela primeira vez no Radar, advertimos que seu uso indevido poderia levar a antipadrões que, a longo prazo, trazem mais desvantagens do que benefícios. No entanto, vimos um interesse crescente no GraphQL entre nossos times devido à sua capacidade de agregar informações de diferentes recursos. Desta vez, queremos alertar sobre o uso de Apollo Federation e seu forte suporte para um único grafo de dados unificado para sua empresa. Embora à primeira vista a ideia de ter conceitos onipresentes em toda a organização seja tentadora, temos que levar em consideração tentativas anteriores semelhantes na indústria — como MDM e modelos de dados canônicos, entre outras — que expuseram as armadilhas desta abordagem. Os desafios podem ser significativos, especialmente quando o domínio em que nos encontramos é muito complexo para se criar um modelo unificado exclusivo.

ESBs disfarçados de gateways de API

Evite

Há muito tempo alertamos contra barramentos de serviços empresariais centralizados (ESBs) e definimos "endpoints inteligentes, pipes burros" como uma das principais características de uma arquitetura de microsserviços. Infelizmente, estamos observando um padrão de ESBs tradicionais passando por processos de rebranding, criando um cenário de ESBs disfarçados de gateways de API, que naturalmente encorajam gateways de API excessivamente ambiciosos. Não se deixe enganar pelo marketing: independentemente do nome que você dê para isso, colocar a lógica de negócio (incluindo orquestração e transformação) em uma ferramenta centralizada cria um acoplamento arquitetural, diminui a transparência e aumenta o aprisionamento a fornecedoras sem vantagens aparentes. Os gateways de API ainda podem atuar como uma abstração útil para questões transversais, mas acreditamos que a inteligência deve residir nas próprias APIs.

Agregação de logs para análise de negócio

Evite

Vários anos atrás, surgiu uma nova geração de plataformas de agregação de logs, capazes de armazenar e pesquisar vastas quantidades de dados de log para descobrir tendências e insights de dados operacionais. Splunk foi o mais proeminente, mas de maneira alguma o único exemplo dessas ferramentas. Como essas plataformas fornecem ampla visibilidade operacional e de segurança em todo o conjunto de aplicações, pessoas desenvolvedoras e administradoras se tornaram cada vez mais dependentes delas. Esse entusiasmo se espalhou à medida que stakeholders descobriram que podiam usar a agregação de logs para análise de negócio. No entanto, as necessidades de negócio podem superar rapidamente a flexibilidade e a usabilidade dessas ferramentas. Os logs destinados à observabilidade técnica geralmente são inadequados para inferir uma compreensão profunda de clientes. Preferimos usar ferramentas e métricas projetadas especificamente para análise de clientes ou adotar uma abordagem mais orientada a eventos para a observabilidade, na qual os eventos comerciais e operacionais são coletados e armazenados de forma que possam ser reproduzidos e processados por ferramentas mais específicas.

Técnicas

Independentemente do nome que você dê para isso, colocar a lógica de negócio em uma ferramenta centralizada cria um acoplamento arquitetural, diminui a transparência e aumenta o aprisionamento a fornecedoras sem vantagens aparentes.

(ESBs disfarçados de gateways de API)

Técnicas

Vemos uma tendência preocupante de usar micro frontends como desculpa para misturar uma variedade de tecnologias, ferramentas ou frameworks concorrentes em uma única página. Embora isso possa ser tecnicamente possível, está longe de ser aconselhável.

(Anarquia de micro frontends)

Anarquia de micro frontends

Evite

Desde que introduzimos originalmente o termo em 2016, os micro frontends cresceram em popularidade e conquistaram aceitação do público em geral. Mas, como qualquer nova técnica com um nome fácil de lembrar, ocasionalmente ela foi mal utilizada. É particularmente preocupante a tendência de usar essa arquitetura como uma desculpa para misturar uma variedade de tecnologias, ferramentas ou frameworks concorrentes em uma única página, levando à anarquia de micro frontends. Um exemplo particularmente ruim dessa síndrome é o uso de vários frameworks de frontend — por exemplo, React.js e Angular — na mesma aplicação de "página única" (SPA). Embora isso possa ser tecnicamente possível, está longe de ser aconselhável quando não faz parte de uma estratégia de transição deliberada. Outras propriedades que devem ser consistentes entre os times incluem técnicas de estilo (por exemplo, CSS-in-JS ou módulos CSS) e os meios pelos quais os componentes individuais são integrados (por exemplo, iFrames ou componentes web). Além disso, as organizações devem decidir se padronizam abordagens consistentemente ou deixam que seus times decidam sobre o gerenciamento de estado, obtenção de dados, ferramentas de compilação, análises e uma série de outras opções em aplicações micro frontend.

Uso de notebooks em produção

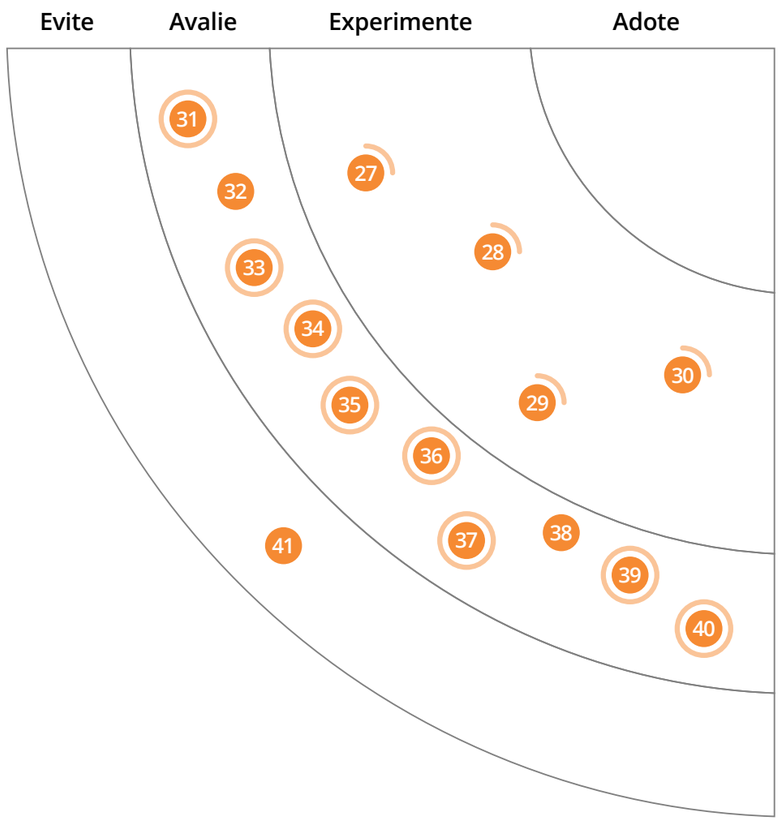
Evite

Nas últimas décadas, notebooks computacionais, introduzidos pela primeira vez pelo Wolfram Mathematica, evoluíram para apoiar pesquisas científicas, explorações e fluxos de trabalho educacionais. Naturalmente, apoiando fluxos de trabalho de ciência de dados e com opções semelhantes aos notebooks Jupyter e Databricks, eles se tornaram grandes companheiros, fornecendo um ambiente de computação interativo, simples e intuitivo para combinação de código para analisar dados com texto rico, e visualização, para contar uma história a partir dos dados. Os notebooks foram projetados para fornecer um meio definitivo para comunicação e inovação científica moderna. Nos últimos anos, entretanto, vimos uma tendência de usar notebooks como meios para executar em produção código com o tipo de qualidade normalmente necessária para conduzir operações corporativas. Vemos fornecedoras de plataformas de notebook anunciando o uso de seus notebooks exploratórios em produção. Este é um exemplo de uma boa intenção — democratizar a programação para cientistas de dados — implementada de forma inadequada, prejudicando a escalabilidade, a manutenção, a resiliência e todas as outras qualidades que um código de produção de longa duração precisa suportar. Não recomendamos o uso de notebooks em produção e, em vez disso, incentivamos a habilitação de cientistas de dados para criar código pronto para produção com os frameworks de programação certos, simplificando assim o conjunto de ferramentas de entrega contínua e removendo a complexidade por meio de plataformas de ML de ponta a ponta.

TECHNOLOGY RADAR

Plataformas

Plataformas



Azure DevOps
Experimente

O Azure DevOps reúne um conjunto de serviços gerenciados, incluindo repositórios Git hospedados na nuvem, pipelines de CI/CD, ferramentas de teste automatizadas, conjunto de ferramentas de gerenciamento de backlog e repositório de artefatos. Vemos nossos times obtendo mais experiência no uso da plataforma, com bons resultados, o que indica que o Azure DevOps está amadurecendo. Gostamos particularmente de sua flexibilidade; ele permite que você use os serviços que deseja, mesmo que sejam de fornecedoras diferentes. Por exemplo, você pode usar um repositório Git externo enquanto ainda usa os serviços de pipeline do Azure DevOps. Nossos times estão particularmente entusiasmados com o Azure DevOps Pipelines. Todos os serviços, no entanto, oferecem uma boa experiência de desenvolvimento que ajuda nossos times a entregar valor.

Debezium
Experimente

Debezium é uma plataforma de captura de dados alternados (CDC) que pode transmitir alterações de banco de dados em tópicos Kafka. CDC é uma técnica popular com vários casos de uso, incluindo a replicação de dados para outros bancos, alimentação de sistemas analíticos, extração de microsserviços de monólitos e invalidação de caches. O Debezium reage às mudanças nos arquivos de log do banco de dados e tem conectores CDC para vários bancos de dados, incluindo Postgres, MySQL, Oracle e MongoDB. Estamos usando o Debezium em muitos projetos e tem nos atendido muito bem.

Honeycomb
Experimente

Honeycomb é um serviço de observabilidade que ingere dados ricos de sistemas de produção e os torna gerenciáveis por meio de amostragem dinâmica. Os times de desenvolvimento podem registrar grandes quantidades de eventos enriquecidos e, posteriormente, decidir como dividi-los e correlacioná-los. Essa abordagem interativa é útil ao trabalhar com os grandes sistemas distribuídos de hoje, já que passamos do ponto em que podemos razoavelmente antecipar quais perguntas podem ser feitas aos sistemas de produção. O time por trás do Honeycomb está desenvolvendo ativamente para uma série de linguagens e frameworks, com plugins disponíveis para Go, Node, Java e Rails, entre outros. Outros novos recursos estão sendo

Adote

Experimente
27. Azure DevOps
28. Debezium
29. Honeycomb
30. JupyterLab

Avalie
31. Amundsen
32. AWS Cloud Development Kit
33. Backstage
34. Dremio
35. DuckDB
36. K3s
37. Materialize
38. Pulumi
39. Tekton
40. Trust over IP stack

Evite
41. Uso excessivo de Node.js

Plataformas

O ambiente interativo do JupyterLab é uma evolução do Jupyter Notebook: ele amplia os recursos originais com células de arrastar e soltar e preenchimento automático de guias, entre outras novas funcionalidades.

(JupyterLab)

O Backstage do Spotify é uma plataforma de código aberto para a criação de portais de desenvolvimento, que pode ajudar a padronizar o número de ferramentas e tecnologias que seus times estão usando e evitar que seu ecossistema de software se torne fragmentado e complexo.

(Backstage)

adicionados em um ritmo rápido. O modelo de precificação também foi simplificado para torná-lo mais atraente. Nossos times têm gostado muito.

JupyterLab
Experimente

Desde a introdução do JupyterLab no anel Avalie em nossa última edição, ele se tornou a interface de usuário web preferida para o Projeto Jupyter entre boa parte do nosso time de profissionais de dados. O uso do JupyterLab está superando rapidamente o uso de Notebooks Jupyter, que eventualmente serão substituídos. Se você ainda estiver usando Notebooks Jupyter, experimente o JupyterLab. Seu ambiente interativo é uma evolução do Notebook Jupyter: ele amplia os recursos originais com células de arrastar e soltar e preenchimento automático de guias, entre outras novas funcionalidades.

Amundsen
Avalie

Cientistas de dados gastam grande parte de seu tempo na descoberta de dados, o que significa que ferramentas que ajudem neste espaço costumam gerar algum entusiasmo. Embora o projeto Apache Atlas tenha se tornado a ferramenta de escolha para gerenciamento de metadados, a descoberta de dados ainda não é facilmente realizada. É aí que entra o Amundsen, que pode ser implantado em conjunto com o Apache Atlas para fornecer uma interface de pesquisa muito mais agradável para descoberta de dados.

AWS Cloud Development Kit
Avalie

Para muitos de nossos times, Terraform se tornou a escolha padrão para definição de infraestrutura de nuvem. No entanto, alguns de nossos times estão experimentando o AWS Cloud Development Kit (AWS CDK) e gostando do que viram até agora. Particularmente, gostam do uso de linguagens de programação de primeira classe em vez de arquivos de configuração, o que lhes permite usar as ferramentas, abordagens de teste e habilidades já existentes. Assim como ferramentas semelhantes, ainda é necessário cuidado para garantir que as implantações permaneçam fáceis de entender e manter. Atualmente, é compatível com TypeScript, JavaScript, Python, Java e C# e .NET. Continuaremos acompanhando o AWS CDK, especialmente considerando que as equipes da AWS e da HashiCorp lançaram recentemente um preview do Cloud Development Kit para Terraform, para gerar configurações do Terraform e permitir o provisionamento com a plataforma.

Backstage
Avalie

As organizações têm buscado apoiar e otimizar os ambientes de desenvolvimento por meio de portais ou plataformas de desenvolvimento. À medida que o número de ferramentas e tecnologias aumenta, uma forma de padronização se torna cada vez mais importante para garantir consistência. Assim, as pessoas desenvolvedoras podem se concentrar na inovação e no desenvolvimento de produtos, em vez de se verem obrigadas a reinventar a roda. Um portal de desenvolvimento centralizado pode oferecer fácil descoberta de serviços e práticas recomendadas. Backstage é uma plataforma de código aberto do Spotify para a criação de portais de desenvolvimento. É baseada em modelos de software, ferramentas de infraestrutura unificadas e documentação técnica consistente e centralizada. Sua arquitetura de plugins permite extensibilidade e adaptabilidade no ecossistema de infraestrutura de uma organização.

Dremio
Avalie

Dremio é uma ferramenta de lago de dados em nuvem que possibilita consultas interativas em armazenamentos de lagos de dados em nuvem. Com o Dremio, você não precisa gerenciar pipelines de dados para extrair e transformar dados em um data warehouse separado para desempenho preditivo. O Dremio cria conjuntos de dados virtuais a partir dos dados ingeridos em um lago de dados, fornecendo uma visão uniforme para o público consumidor. O Presto popularizou a técnica de separar o armazenamento da camada de computação, e o Dremio leva isso adiante, melhorando o desempenho e otimizando o custo de operação.

DuckDB
Avalie

DuckDB é um banco de dados colunar incorporado para ciência de dados

e cargas de trabalho analíticas. Analistas gastam muito tempo limpando e visualizando os dados localmente antes de enviá-los para os servidores. Embora os bancos de dados existam há décadas, a maioria deles é projetada para casos de uso cliente-servidor e, portanto, não adequados para consultas interativas locais. Para contornar essa limitação, analistas geralmente recorrem a ferramentas de processamento de dados na memória, como Pandas ou data.table. Embora essas ferramentas sejam eficazes, elas limitam o escopo da análise ao volume de dados que cabe na memória. Sentimos que o DuckDB preenche perfeitamente essa lacuna entre ferramentas, com um mecanismo colunar integrado que é otimizado para análises em conjuntos de dados locais maiores que a capacidade da memória.

K3s
Avalie

O K3s é uma distribuição leve do Kubernetes desenvolvida para IoT e computação de borda. É empacotado como um único binário e tem dependências mínimas ou nenhuma de sistemas operacionais, tornando-o realmente fácil de operar e usar. Ele usa o sqlite3 como armazenamento padrão em vez do etcd. O K3s tem uma pegada de memória reduzida porque executa todos os componentes relevantes em um único processo. Ele também atinge um binário menor ao remover drivers de armazenamento de terceiras e provedoras de nuvem que não são relevantes para os casos de uso do K3s. Para ambientes com recursos limitados, K3s é uma escolha muito boa e vale a pena considerar.

Materialize
Avalie

Materialize é um banco de dados de streaming que permite fazer computação incremental sem pipelines de dados complicados. Basta descrever seus cálculos por meio de visualizações SQL padrão e conectar o Materialize ao fluxo de dados. A ferramenta subjacente de fluxo de dados diferencial executa computação incremental para fornecer saída consistente e correta com latência mínima. Ao contrário dos bancos de dados tradicionais, não há restrições na definição dessas visualizações materializadas e os cálculos são executados em tempo real.

Pulumi
Avalie

Temos visto o interesse no Pulumi crescer de forma lenta, mas contínua.
O Pulumi preenche uma lacuna no mundo da programação de infraestrutura, no qual o Terraform mantém uma posição sólida. Embora o Terraform seja uma ferramenta testada e aprovada, sua natureza declarativa sofre com recursos de abstração inadequados e testabilidade limitada. O Terraform é adequado quando a infraestrutura é totalmente estática, mas as definições de infraestrutura dinâmica exigem uma linguagem de programação real. Pulumi se distingue por permitir que as configurações sejam escritas em TypeScript/JavaScript, Python e Go — sem necessidade de linguagem de marcação ou modelagem. O Pulumi tem foco fortemente voltado para arquiteturas nativas da nuvem — incluindo contêineres, funções sem servidor e serviços de dados — e oferece bom suporte para Kubernetes. Recentemente, o AWS CDK surgiu como desafiante, mas Pulumi continua a ser a única ferramenta neutra em nuvem nesta área. Estamos antecipando uma adoção mais ampla do Pulumi no futuro e otimistas por uma ferramenta viável e um ecossistema de conhecimento emergente para apoiá-lo.

Tekton
Avalie

Tekton é uma jovem plataforma Kubernetes nativa para gerenciamento de pipelines de integração e entrega contínua (CI/CD). Tekton não apenas instala e executa no Kubernetes, como também define seus pipelines de CI/CD como recursos personalizados do Kubernetes. Isso significa que os pipelines podem ser controlados por clientes nativos do Kubernetes (CLI ou APIs), e podem se beneficiar de funcionalidades de gerenciamento de recursos subjacentes, como reversões. O formato de declaração do pipeline é flexível e permite definir fluxos de trabalho com condições, caminhos de execução paralela e manipulação de tarefas finais para limpeza, entre outros recursos. Como resultado, o Tekton pode suportar fluxos de trabalho de implantação complexos e híbridos com reversões, implantações canário e muito mais. Tekton é uma plataforma de código aberto e também é fornecido como um serviço gerenciado pelo GCP. Embora a documentação tenha espaço para melhorias e a comunidade esteja crescendo, temos usado o Tekton com sucesso para cargas de trabalho de produção na AWS.

Plataformas

K3s é uma distribuição leve do Kubernetes desenvolvida para IoT e computação de borda, que remove drivers de armazenamento de terceiras e provedoras de nuvem que não são relevantes para esses casos de uso.

(K3s)

Materialize é um banco de dados de streaming que permite fazer cálculos incrementais sem pipelines de dados complicados.

(Materialize)

Plataformas

Esta stack de quatro camadas de tecnologia e governança visa estabelecer um patamar para uma forma altamente interoperável de gerenciamento de identidade descentralizada.

(Trust over IP stack)

O Node.js é extremamente popular. Mas isso não o torna adequado para tudo — é uma escolha ruim, por exemplo, para cargas de trabalho pesadas em computação. Nós advertimos contra a tendência de usar o Node.js indiscriminadamente ou pelos motivos errados.

(Uso excessivo de Node.js)

Trust over IP stack

Avalie

Os desafios contínuos relacionados a como indivíduos e organizações estabelecem confiança digitalmente, pela Internet, estão dando origem a uma nova abordagem para provar identidade, compartilhar e verificar os atributos necessários para estabelecer confiança e fazer transações com segurança. Nesta edição do Radar, destacamos algumas das tecnologias fundamentais, como identidade descentralizada e credenciais verificáveis, que viabilizam esta nova era de confiança digital. No entanto, uma mudança dessa dimensão, em escala global, não será possível sem a padronização de uma stack de governança técnica que habilite a interoperabilidade. A nova Fundação Trust over IP, parte da Fundação Linux, se propôs a fazer exatamente isso. Inspirando-se na forma como a padronização TCP/IP como protocolo universal da Internet permitiu a interoperabilidade entre bilhões de dispositivos, o grupo está definindo uma arquitetura de quatro camadas de tecnologia e governança, por meio da Trust over IP stack. A stack inclui utilitários públicos (como identificadores descentralizados), comms de identidade descentralizada para protocolos padronizados para agentes (como carteiras digitais para comunicação), protocolos de troca de dados (como fluxos para emitir e verificar credenciais verificáveis), bem como ecossistemas de aplicações (como educação, finanças, saúde, entre outros). Se você está revisando seus sistemas de identidade e a forma como estabelece confiança com seu ecossistema, sugerimos conferir a stack ToIP e seu conjunto de ferramentas de suporte, Hyperledger Aries.

Uso excessivo de Node.js

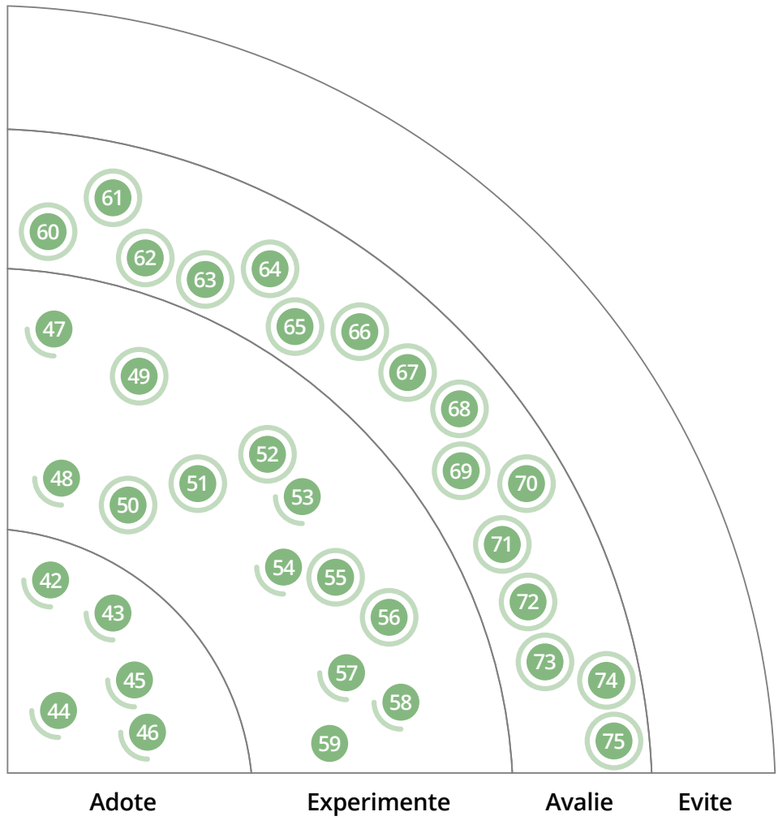
Evite

As tecnologias, especialmente as extremamente populares, tendem a ser usadas excessivamente. O que estamos vendo no momento é o uso excessivo de Node.js, uma tendência a usar o Node.js indiscriminadamente ou pelos motivos errados. Entre estes, dois se destacam em nossa opinião. Primeiramente, ouvimos frequentemente que o Node deve ser usado para que toda a programação possa ser feita em apenas uma linguagem de programação. Nossa opinião é de que a programação poliglota é uma abordagem melhor, e isso vale nos dois sentidos. Em segundo lugar, geralmente ouvimos times citando o desempenho como uma razão para escolher o Node.js. Embora existam inúmeros benchmarks mais ou menos razoáveis, essa percepção está enraizada na história. Quando o Node.js se tornou popular, foi o primeiro grande framework a adotar um modelo de programação de não-bloqueio, que o tornou muito eficiente para tarefas pesadas de IO (mencionamos isso na redação do blip do Node.js. em 2012). Devido à sua natureza de thread única, o Node.js. nunca foi uma boa opção para cargas de trabalho pesadas em computação, e agora que existem frameworks não-bloqueadores eficientes em outras plataformas — alguns com APIs elegantes e modernas — o desempenho não é mais um motivo para escolher o Node.js.

TECHNOLOGY RADAR

Ferramentas

Ferramentas



Airflow

Adote

Airflow continua sendo nossa ferramenta de gerenciamento de fluxo de trabalho de código aberto favorita para pipelines de processamento de dados como gráficos acíclicos direcionados (DAGs). Este é um espaço em crescimento, com ferramentas de código aberto, como Luigi e Argo, e ferramentas específicas de fornecedoras, como Azure Data Factory e AWS Data Pipeline. No entanto, o Airflow se diferencia com sua definição programática de fluxos de trabalho sobre arquivos de configuração de baixo código limitados, suporte para testes automatizados, instalação multiplataforma de código aberto, um conjunto rico de pontos de integração para o ecossistema de dados e grande suporte da comunidade. Em arquiteturas de dados descentralizadas, como malha de dados, no entanto, o Airflow atualmente falha como uma orquestração de fluxo de trabalho centralizado.

Bitrise

Adote

Bitrise, uma ferramenta de CD de domínio específico para aplicações móveis, continua sendo uma parte útil do fluxo de trabalho móvel, e os times realmente deveriam usá-la. Bitrise permite construir, testar e implantar aplicações móveis do laptop da pessoa desenvolvedora até a publicação na loja de aplicativos. É fácil de configurar e fornece um conjunto abrangente de etapas predefinidas para a maioria das necessidades de desenvolvimento móvel.

Dependabot

Adote

Entre as ferramentas disponíveis para manter as dependências atualizadas, Dependabot é uma escolha padrão sólida em nossa opinião. A integração do Dependabot com o GitHub é suave e as solicitações de pull são enviadas automaticamente para atualizar suas dependências para as versões mais recentes. Ele pode ser habilitado no nível da organização, portanto, é muito simples para os times receber essas solicitações pull. Se você não estiver usando o GitHub, ainda poderá usar as bibliotecas Dependabot em seu pipeline

Adote
42. Airflow
43. Bitrise
44. Dependabot
45. Helm
46. Trivy

Experimente
47. Bokeh
48. Concourse
49. Dash
50. jscodeshift
51. Kustomize
52. MLflow
53. Pitest
54. Sentry
55. ShellCheck
56. Stryker
57. Terragrunt
58. tfsec
59. Yarn

Avalie
60. CML
61. Eleventy
62. Flagger
63. gossm
64. Great Expectations
65. k6
66. Katran
67. Kiali
68. LGTM
69. Litmus
70. Opacus
71. OSS Index
72. Playwright
73. pnpm
74. Sensei
75. Zola

Evite

Ferramentas

Trivy é um scanner de vulnerabilidade para contêineres, que vem como um binário autônomo, facilitando a configuração e execução do escaneamento localmente.

(Trivy)

de compilação. Se você tiver interesse em uma ferramenta alternativa, considere também o Renovate, que oferece suporte a uma gama mais ampla de serviços, incluindo GitLab, Bitbucket and Azure DevOps.

Helm
Adote

Helm é um gerenciador de pacotes para Kubernetes. Ele vem com um repositório de aplicações Kubernetes selecionadas que são mantidas no repositório de gráficos oficial. Desde que falamos sobre o Helm pela última vez, o Helm 3 foi lançado e a mudança mais significativa é a remoção do Tiller, o componente do lado do servidor do Helm 2. A vantagem de um design sem o Tiller é que você só pode fazer alterações no cluster do Kubernetes do lado do cliente, ou seja, você só pode modificar o cluster de acordo com as permissões que possui como usuário do comando Helm. Usamos Helm em vários projetos de clientes e seu gerenciamento de dependências, modelos e mecanismo de hook simplificou muito o gerenciamento do ciclo de vida de aplicações no Kubernetes.

Trivy
Adote

Pipelines de compilação que criam e implantam contêineres devem incluir escaneamento de segurança de contêiner. Nossos times gostam particularmente do Trivy, um scanner de vulnerabilidade para contêineres. Nós testamos Clair e Anchore Engine, entre outras boas ferramentas neste campo. Ao contrário do Clair, o Trivy não verifica apenas contêineres, mas também dependências na base de código. Além disso, como o Trivy é fornecido como um binário autônomo, é mais fácil configurar e executar o escaneamento localmente. Outros benefícios do Trivy incluem o fato de ser um software de código aberto e suportar contêineres sem distribuição.

Bokeh
Experimente

Bokeh é uma das principais bibliotecas em Python para a criação de gráficos científicos e visualizações de dados renderizados no navegador via JavaScript. Essas ferramentas, em comparação com as ferramentas de desktop que criam imagens estáticas, facilitam a reutilização de código para trabalho exploratório em aplicações web. Bokeh é particularmente útil para isso. A biblioteca é madura e cheia de recursos. O que gostamos em Bokeh é que ela é ótima em manter sua preocupação como uma ferramenta de camada de apresentação, e não tentar assumir preocupações como agregação de dados (consulte ggplot) ou desenvolvimento de aplicações web (como Shiny ou Dash). Por isso, é uma ótima opção para usar quando a separação de interesses for importante para você. Bokeh fornece widgets de IU web e pode ser executada no modo de servidor, mas você pode usar ou descartar esses recursos como achar melhor. Bokeh é flexível e não faz muitas suposições sobre como você vai usá-la, nem tem muitas dependências (como pandas ou notebooks).

Concourse
Experimente

A implementação de pipelines de entrega contínua sustentáveis que podem compilar e implantar software de produção em vários ambientes requer uma ferramenta que trate pipelines de compilação e artefatos como cidadãos de primeira classe. Quando começamos a avaliar o Concourse, gostamos de seu modelo simples e flexível, o princípio de compilações baseadas em contêiner e o fato de que ele obriga você a definir pipelines como código. Desde então, a usabilidade melhorou e o modelo simples resistiu ao teste do tempo. Muitos de nossos times e clientes têm usado com sucesso o Concourse para a configuração de pipelines grandes por longos períodos de tempo. Com frequência, também nos beneficiamos da flexibilidade do Concourse para executar workers em qualquer lugar, por exemplo, quando os testes de integração de hardware exigem uma configuração local.

Dash
Experimente

Esta edição do Radar apresenta várias novas ferramentas para a criação de aplicações web que ajudam usuários finais a visualizar e interagir com dados. Essas ferramentas vão além de bibliotecas de visualização simples, como D3. Em vez disso, elas reduzem o esforço necessário para construir aplicações analíticas independentes para manipular conjuntos de dados existentes. Dash, da Plotly, está ganhando popularidade entre cientistas de dados por criar aplicações analíticas amplamente funcionais em Python. O Dash aumenta as bibliotecas de dados do Python, de forma semelhante ao que Shiny faz a partir do R. Essas aplicações às vezes são chamadas de painéis, mas a gama de funcionalidades possíveis é realmente muito maior do que o termo implica. Dash é particularmente adequado para criar aplicações escaláveis e prontas para produção, ao contrário do Streamlit, outra ferramenta desta classe. Considere o uso do Dash quando precisar apresentar aos usuários de negócios análises mais sofisticadas do que uma solução com pouco ou nenhum código, como o Tableau, pode fornecer.

jscodeshift
Experimente

A manutenção de bases de código JavaScript em grande escala nunca é fácil, mas é especialmente desafiadora quando migramos alterações que podem gerar quebras. IDEs com recursos de refatoração podem ajudar em cenários simples. No entanto, quando sua base de código é uma biblioteca com ampla dependência, toda vez que você faz uma alteração importante, é preciso passar por uma série de bases de código de clientes para fazer as atualizações apropriadas — o que requer supervisão humana e precisa ser feito de forma manual. jscodeshift, um kit de ferramentas para refatorar JavaScript e TypeScript, ajuda a aliviar essa dor. Ele pode analisar seu código para construir árvores de sintaxe abstratas (AST) e fornecer uma API para manipular a árvore com várias transformações (por exemplo, adicionar, renomear e excluir propriedades de componentes existentes), em seguida exportando a árvore como código-fonte final. jscodeshift também vem com um utilitário de testes de unidade simples que pode aplicar desenvolvimento orientado a testes para escrever codemods de migração. Consideramos o jscodeshift bastante útil para a manutenção de sistemas de design.

Kustomize
Experimente

Kustomize é uma ferramenta para gerenciar e personalizar arquivos de manifesto do Kubernetes. Ele permite selecionar e ajustar recursos básicos do Kubernetes antes de aplicá-los a ambientes diferentes, e agora conta com suporte nativo do kubectl. Gostamos da ferramenta porque ela ajuda a manter seu código DRY e, ao contrário do Helm, (que tenta fazer muitas coisas ao mesmo tempo — gerenciamento de pacotes, gerenciamento de versões, etc.), consideramos que o Kustomize segue a filosofia Unix: faça uma coisa bem feita e espere que a saída de cada programa seja entrada para outro.

MLflow
Experimente

MLflow é uma ferramenta de código aberto para rastreamento de experimentos de aprendizado de máquina e gerenciamento de ciclo de vida. O fluxo de trabalho para desenvolver e evoluir continuamente um modelo de aprendizado de máquina inclui uma série de experimentos (uma coleção de execuções), rastreamento de desempenho desses experimentos (uma coleção de métricas) e rastreamento e ajuste de modelos (projetos).
O MLflow facilita esse fluxo de trabalho muito bem, suportando os padrões abertos existentes e se integrando bem com muitas outras ferramentas no ecossistema. O MLflow como um serviço gerenciado por Databricks na nuvem, disponível em AWS e Azure, está amadurecendo rapidamente e temos usado com sucesso em nossos projetos. Consideramos o MLflow uma ótima ferramenta para gerenciamento e rastreamento de modelos, com suporte para modelos de interação baseados em UI e API. Nossa única e crescente preocupação é que o MLflow tenta entregar muitas questões conflitantes como uma única plataforma, por exemplo, modelos de serviço e pontuação.

Ferramentas

jscodeshift facilita a difícil tarefa de manter bases de código JavaScript em grande escala. Achamos a ferramenta particularmente útil para manutenção de sistemas de design.

(jscodeshift)

Kustomize é uma ferramenta para gerenciar e personalizar arquivos de manifesto do Kubernetes, que permite selecionar e corrigir recursos básicos do Kubernetes antes de aplicá-los a diferentes ambientes

(Kustomize)

Ferramentas

Sentry é uma ferramenta de monitoramento de aplicações multiplataforma com foco em relatórios de erros. Tem ajudado nossos times a rastrear erros, estabelecer se um commit realmente corrigiu um problema e alertar se um problema reaparecer devido a uma regressão.

(Sentry)

Pitest
Experimente

As abordagens de testes tradicionais se concentram em avaliar se nosso código de produção está fazendo o que deveria. No entanto, podemos cometer erros no código de teste, introduzindo afirmações incompletas ou inúteis que criam uma falsa sensação de confiança. É aqui que entra o teste de mutação: ele avalia a qualidade dos próprios testes, encontrando pontos cegos que são difíceis de perceber. Nossos times têm usado Pitest há algum tempo, e recomendamos seu uso em projetos Java para medir a integridade do conjunto de testes. Resumindo, o teste de mutação introduz mudanças no código de produção e executa os mesmos testes uma segunda vez; se os testes ainda estiverem verdes, significa que os testes não estão bons e precisam ser melhorados. Se você estiver usando linguagens de programação diferentes de Java, Stryker é uma boa escolha neste espaço.

Sentry
Experimente

Sentry é uma ferramenta de monitoramento de aplicações multiplataforma com foco em relatórios de erros. Ferramentas como Sentry se diferenciam das soluções de registro tradicionais, como ELK Stack, em seu foco na descoberta, investigação e correção de erros. Sentry já existe há algum tempo e oferece suporte a várias linguagens e frameworks. Usamos Sentry em muitos projetos e tem sido muito útil para rastrear erros, descobrindo se um commit realmente corrigiu um problema e nos alertando se um problema reaparecer devido a uma regressão.

ShellCheck
Experimente

Ainda que as ferramentas tenham evoluído muito no espaço da infraestrutura, escrever um shell script pode fazer sentido em alguns casos. Naturalmente, a sintaxe dos shell scripts é dominada por poucas pessoas e, como temos menos prática escrevendo shell scripts atualmente, passamos a gostar do ShellCheck, um linter para shell scripts. ShellCheck pode ser usado a partir da linha de comando, como parte de uma compilação ou, melhor ainda, como uma extensão em muitos IDEs populares. O wiki contém uma descrição detalhada de centenas de problemas que o ShellCheck pode detectar, e a maioria das ferramentas e IDEs fornecem uma maneira conveniente de acessar a respectiva página do wiki quando um problema é encontrado.

Stryker
Experimente

Stryker é um relativamente novo estreante no espaço de testes de mutação. Semelhante ao Pitest, o Stryker permite avaliar a qualidade dos seus testes. Temos o usado com muito sucesso em projetos de JavaScript, mas ele também oferece suporte a projetos de C# e Scala. O Stryker é muito amigável e altamente personalizável, e temos conseguido aumentar a cobertura de código e a confiança nas aplicações que entregamos para nossas clientes.

Terragrunt
Experimente

Usamos o Terraform extensivamente para criar e gerenciar infraestrutura em nuvem. Em nossa experiência com configurações maiores, nas quais o código é dividido em módulos que são incluídos de maneiras diferentes, os times acabam se deparando com uma parede de repetição inevitável causada por falta de flexibilidade. Lidamos com esse problema usando Terragrunt, um wrapper fino para Terraform que implementa as práticas defendidas por Yevgeniy Brikman em Terraform: Up and Running. Consideramos o Terragrunt útil por encorajar módulos versionados e reutilização para diferentes ambientes. Os ganchos de ciclo de vida são outro recurso útil que fornece flexibilidade adicional. Em termos de empacotamento, o Terragrunt tem as mesmas limitações do Terraform: não há uma maneira adequada de definir pacotes ou dependências entre pacotes. Como solução alternativa, você pode usar módulos e especificar uma versão associada a uma tag Git.

tfsec
Experimente

A segurança é uma preocupação geral e capturar riscos cedo é sempre melhor do que enfrentar problemas mais tarde. No espaço de infraestrutura como código, no qual Terraform é uma escolha óbvia para gerenciar ambientes em nuvem, agora também temos tfsec, uma ferramenta de análise estática que escaneia os modelos do Terraform para encontrar possíveis problemas de segurança. Nossos times

têm usado tfsec com bastante sucesso. A ferramenta é fácil de configurar e usar, o que a torna uma ótima escolha para qualquer time de desenvolvimento determinado a mitigar riscos de segurança para evitar violações. Suas regras predefinidas para diferentes provedores de nuvem, incluindo AWS e Azure, complementam os benefícios que o tfsec traz para times que usam Terraform.

Yarn
Experimente

Yarn continua a ser o gerenciador de pacotes de escolha para muitos times. Recebemos com entusiasmo o lançamento do Yarn 2, um novo release importante, com uma longa lista de mudanças e melhorias. Além de ajustes de usabilidade e melhorias na área de espaços de trabalho, o Yarn 2 introduz o conceito de zero-installs, que permite às pessoas desenvolvedoras executar um projeto diretamente após cloná-lo. No entanto, o Yarn 2 inclui algumas mudanças importantes que tornam a atualização não-trivial. Ele também traz ambientes plug'n'play (PnP) como padrão, mas não oferece suporte ao React Native em ambientes PnP. Os times podem, é claro, desabilitar o PnP ou permanecer no Yarn 1. No entanto, eles devem estar cientes de que o Yarn 1 está agora em modo de manutenção.

CML
Avalie

Incluímos entrega contínua para aprendizado de máquina como uma técnica em edições anteriores do Radar e, nesta edição, queremos destacar uma nova ferramenta promissora chamada Continuous Machine Learning (ou CML), das mesmas pessoas responsáveis pelo DVC. O CML visa trazer as melhores práticas de engenharia de CI e CD para times de IA e ML, e pode ajudar a organizar sua infraestrutura MLOps em cima de uma stack de engenharia de software tradicional, em vez de criar plataformas de IA separadas. Gostamos do fato de priorizarem o suporte para DVC e vemos isso como um bom sinal para essa nova ferramenta em crescimento.

Eleventy
Avalie

Há muito tempo gostamos da ideia de usar geradores de site estáticos para evitar a complexidade e melhorar o desempenho, sempre que o caso de uso permitir. Embora Eleventy já exista há alguns anos, a ferramenta chamou nossa atenção recentemente, à medida que amadureceu e os favoritos anteriores, como Gatsby.js, apresentaram alguns problemas de escalabilidade. Eleventy permite aprendizado rápido e facilidade para a construção de sites. Também gostamos da facilidade para criar marcação semântica (e, portanto, mais acessível) com seus modelos e seu suporte simples e robusto para paginação.

Flagger
Avalie

Malhas de serviços e gateways de API fornecem uma maneira conveniente de rotear o tráfego para uma variedade de microsserviços que implementam a mesma interface de API. Flagger usa esse recurso para ajustar dinamicamente a parte do tráfego que é roteada para uma nova versão de um serviço. Esta é uma técnica comum para implantações canário ou implantações azul-verde. Flagger trabalha em conjunto com uma variedade de proxies populares (incluindo Envoy e Kong) para aumentar progressivamente as solicitações para um serviço e relatar métricas sobre a carga para fornecer feedback rápido sobre uma nova

Ferramentas

Flagger é uma ferramenta útil para ajustar a parte do tráfego que é roteada para uma nova versão de um serviço — o que é útil ao trabalhar com malhas de serviço e gateways de API.

(Flagger)

Ferramentas

O framework Great Expectations fornece governança de dados automatizada, permitindo criar controles integrados que sinalizam anomalias ou problemas de qualidade em pipelines de dados.

(Great Expectations)

implantação. Gostamos da característica do Flagger de simplificar essa prática valiosa para que ela possa ser adotada de forma mais ampla. Embora Flagger seja patrocinado pela Weaveworks, é uma ferramenta independente, sem qualquer obrigação de ser usada em conjunto com outras ferramentas da Weaveworks.

gossm

Avalie

Ao conectar-se a instâncias de servidor na AWS, é recomendável passar por um host bastião em vez de uma conexão direta. No entanto, provisionar um host bastião apenas para essa finalidade pode ser frustrante, e é por isso que o Gerenciador de Sessões do AWS Systems Manager fornece tunelamento para uma conexão mais confortável para seus servidores. gossm é uma ferramenta CLI de código aberto que torna o uso do Gerenciador de Sessões ainda mais conveniente. O gossm permite que você se beneficie da segurança fornecida pelo Gerenciador de Sessões e das políticas IAM do seu terminal usando ferramentas como ssh e scp. Ele também tem alguns recursos ausentes na AWS CLI, incluindo descoberta de servidor e integração SSH.

Great Expectations

Avalie

Com o crescimento de CD4ML, os aspectos operacionais da engenharia e ciência de dados têm recebido mais atenção. A governança de dados automatizada é um aspecto desse desenvolvimento. Great Expectations é um framework que permite criar controles integrados que sinalizam anomalias ou problemas de qualidade em pipelines de dados. Assim como os testes de unidade são executados em um pipeline de compilação, o Great Expectations faz afirmações durante a execução de um pipeline de dados. Isso é útil não apenas para implementar uma espécie de Andon para pipelines de dados, mas também para garantir que algoritmos baseados em modelo permaneçam dentro da faixa operacional determinada por seus dados de treinamento. Controles automatizados como esses podem ajudar a distribuir e democratizar o acesso e a custódia dos dados. O Great Expectations também vem com uma ferramenta de criação de perfil para ajudar a entender as qualidades de um determinado conjunto de dados e definir os limites apropriados.

k6

Avalie

Vemos com muito entusiasmo o k6, uma ferramenta relativamente nova no ecossistema de testes de desempenho, com um forte foco na experiência de desenvolvimento. O executor de linha de comando do k6 executa scripts escritos em JavaScript e permite configurar o tempo de execução e o número de usuários virtuais. A CLI tem vários recursos avançados que permitem visualizar as estatísticas atuais antes que o teste termine de ser executado, aumentar o número de usuários virtuais além do que foi definido originalmente e até mesmo pausar e retomar um teste em execução. A saída da linha de comando fornece um conjunto de métricas personalizáveis com transformadores que permitem visualizar os resultados em Datadog e outras ferramentas de observabilidade. Adicionar checks aos seus scripts é uma maneira fácil de integrar o teste de desempenho em seu pipeline de CI/CD. Para testes de desempenho acelerados, confira a versão comercial, k6 Cloud,que fornece escalonamento de nuvem e visualizações adicionais.

Katran

Avalie

Katran é um balanceador de carga de camada 4 de alto desempenho. Não é para todo mundo, mas se você precisar de redundância para balanceadores de carga de camada 7 (como HAProxy ou NGINX), ou precisar escalar balanceadores de carga para dois ou mais servidores, então recomendamos avaliar o Katran. Vemos o Katran como uma escolha mais flexível e eficiente em relação a técnicas como round-robin DNS sobre balanceadores de carga de camada 7, ou o modelo de Kernel IPVS que profissionais de engenharia de rede geralmente adotam para resolver desafios semelhantes.

Kiali

Avalie

Dado o aumento do uso de malha de serviços para implantar coleções de microsserviços em contêineres, podemos esperar o surgimento de ferramentas que automatizem e simplifiquem as tarefas administrativas associadas a este

estilo de arquitetura. Kiali é uma dessas ferramentas. Kiali fornece uma interface de usuário gráfica para observar e controlar redes de serviços implantados com Istio. Achamos Kiali útil para visualizar a topologia de serviços em uma rede e entender o tráfego roteado entre eles. Por exemplo, quando usado em conjunto com Flagger, o Kiali pode exibir solicitações que foram roteadas para uma implantação de serviço canário. Gostamos particularmente da capacidade do Kiali de injetar artificialmente falhas de rede em uma malha de serviços para testar sua resiliência diante de interrupções na rede. Essa prática é frequentemente ignorada devido à complexidade de configurar e executar testes de falha em uma malha complexa de microsserviços.

LGTM
Avalie

Escrever código seguro sempre foi importante, mas é apenas uma das muitas coisas que as pessoas desenvolvedoras devem priorizar. O LGTM fornece uma rede de segurança e um meio para se beneficiar de uma base de conhecimento de práticas de programação seguras. É uma ferramenta de análise de código estático com foco na segurança, apoiada por um catálogo (parcialmente de código aberto) de regras de programação seguras. As regras são implementadas como consultas em sua base de código na linguagem de consulta CodeQL. A ferramenta pode ser usada para integrar verificações de segurança de caixa branca em seus pipelines de CD para Java, Go, JavaScript, Python, C # e C/C++. LGTM e CodeQL são parte do Github Security Lab.

Litmus
Avalie

Litmus é uma ferramenta de engenharia do caos com uma baixa barreira de entrada. Ele permite injetar vários cenários de erro em seu cluster Kubernetes com o mínimo de esforço. Tem nos impressionado positivamente a variedade de recursos que o Litmus oferece para além da eliminação aleatória de pod, incluindo simulação de problemas de rede, CPU, memória e E/S. O Litmus também oferece suporte a experimentos personalizados para simular erros para Kafka e Cassandra, entre outros serviços comuns.

Opacus
Avalie

O conceito de privacidade diferencial apareceu pela primeira vez no Radar em 2016. Embora o problema de quebrar a privacidade por meio de consultas de inferência de modelo sistemáticas tenha sido reconhecido naquela época, era majoritariamente uma questão teórica, já que existiam poucas soluções práticas. A indústria carece de ferramentas para evitar que isso aconteça. Opacus é uma nova biblioteca Python que pode ser usada em conjunto com PyTorch para ajudar a impedir um tipo de ataque de privacidade diferencial. Embora seja um desenvolvimento promissor, encontrar o modelo e o conjunto de dados certos para sua aplicação tem sido um desafio. A biblioteca ainda é bastante nova, por isso estamos na expectativa para observar como será sua aceitação no futuro.

OSS Index
Avalie

É importante para um time de desenvolvimento identificar se as dependências de sua aplicação têm vulnerabilidades conhecidas. O OSS Index pode ser usado para esse objetivo. O OSS Index é um catálogo gratuito de componentes e ferramentas de escaneamento de código aberto, projetados para ajudar os times de desenvolvimento a identificar vulnerabilidades, entender os riscos e manter seu software seguro. Nossos times já estão integrando esse index em pipelines por meio de diferentes

Ferramentas

LGTM é uma ferramenta de análise de código estático com foco na segurança, apoiada por um catálogo de regras de programação seguras.

(LGTM)

Ferramentas

Sensei facilita a criação e distribuição de diretrizes de qualidade de código seguro.

(Sensei)

linguagens, incluindo AuditJS e Gradle plugin. A velocidade é alta, as vulnerabilidades são identificadas com precisão e há pouca ocorrência de falsos positivos.

Playwright
Avalie

O espaço de testes de IU web continua ativo. Algumas das pessoas responsáveis pela criação do Puppeteer se juntaram à Microsoft e agora estão aplicando seus aprendizados no Playwright, que permite escrever testes para Chromium, Firefox, e também para WebKit, tudo por meio da mesma API. O Playwright ganhou atenção por seu suporte a todos os principais navegadores, algo que ele consegue fazer atualmente, incluindo versões patched do Firefox e Webkit. Resta saber o quão rapidamente outras ferramentas vão conseguir se equiparar, com suporte cada vez maior para o Chrome DevTools Protocol como uma API comum para automatizar navegadores.

pnpm
Avalie

pnpm é um gerenciador de pacotes para Node.js promissor, que estamos examinando de perto por sua velocidade e eficiência superiores em comparação com outros gerenciadores de pacotes. As dependências são salvas em um único lugar no disco e vinculadas aos respectivos diretórios node_modules. pnpm também oferece suporte à otimização incremental no nível do arquivo, fornece uma API base sólida para permitir extensão/personalização e oferece suporte ao modo de servidor de armazenamento, o que acelera ainda mais o download de dependências. Se sua organização tem um grande número de projetos com as mesmas dependências, você pode se interessar por examinar o pnpm mais de perto.

Sensei
Avalie

Sensei, do Secure Code Warrior, é um plugin Java IDE que facilita a criação e a distribuição de diretrizes de qualidade de código seguro. Na ThoughtWorks, muitas vezes defendemos “ferramentas acima de regras”, ou seja, facilitar que a coisa certa seja feita em vez de aplicar regras e procedimentos de governança como checklists. E essa ferramenta se encaixa nessa filosofia. As pessoas desenvolvedoras podem criar receitas facilmente compartilhadas com outros membros do time. Elas podem ser simples ou complexas, e são implementadas como consultas direcionadas para Java AST. Os exemplos incluem avisos para injeção de SQL, fraqueza criptográfica e muitos outros. Outro recurso que gostamos: por ser executado em alterações de código no IDE, o Sensei fornece feedback mais rápido do que as ferramentas de análise estática mais tradicionais.

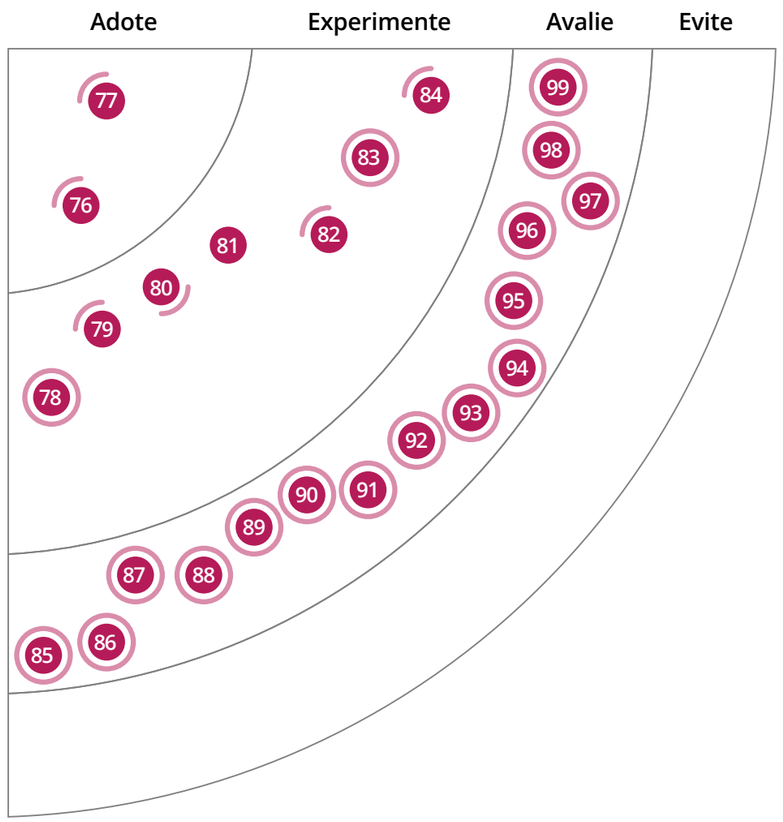
Zola
Avalie

Zola é um gerador de site estático escrito em Rust. Como tal, ele vem como um único executável sem dependências, é muito rápido e suporta tudo o que é esperado, como Sass, conteúdo em markdown e recarregamento a quente. Tivemos sucesso na construção de sites estáticos com o Zola e apreciamos sua usabilidade intuitiva.

TECHNOLOGY RADAR

Linguagens & Frameworks

Linguagens & Frameworks



Adote
76. Arrow
77. jest-when

Experimente
78. Fastify
79. Immer
80. Redux
81. Rust
82. single-spa
83. Strikt
84. XState

Avalie
85. Babylon.js
86. Blazor
87. Flutter Driver
88. HashiCorp Sentinel
89. Hermes
90. io-ts
91. Kedro
92. LitElement
93. Mock Service Worker
94. Recoil
95. Snorkel
96. Streamlit
97. Svelte
98. SWR
99. Testing Library

Evite

Arrow
Adote

Arrow é promovida como a companheira funcional da biblioteca padrão de Kotlin. Na verdade, o pacote de abstrações de alto nível e prontas para uso fornecido por Arrow provou ser tão útil que nossos times agora a consideram um padrão sensato ao trabalhar com Kotlin. Recentemente, em preparação para o release 1.0, a equipe da Arrow introduziu várias mudanças, incluindo a adição de novos módulos, mas também algumas descontinuações e remoções.

jest-when
Adote

jest-when é uma biblioteca JavaScript leve que complementa o Jest combinando argumentos de chamada de função simulada. Jest é uma ótima ferramenta para testar a stack; jest-when permite que você espere por argumentos específicos para funções simuladas, habilitando você a escrever testes de unidade mais robustos de módulos com muitas dependências. É fácil de usar e oferece ótimo suporte para vários matchers, e é por isso que nossos times adotaram jest-when como escolha padrão para simulações neste espaço.

Fastify
Experimente

Nos casos em que a implementação em Node.js é necessária, observamos que Fastify é uma opção que deixa nossos times bastante satisfeitos. Este framework web oferece facilidade no manuseio de validações de requisição-resposta, suporte para TypeScript e um ecossistema de plugins que proporciona aos nossos times uma experiência mais simples de desenvolvimento de software. Embora seja uma boa opção no ecossistema Node.js, mantemos nosso conselho anterior: não caia em cenários de uso excessivo de Node.

Immer
Experimente

Com a complexidade crescente das aplicações de página única (SPA) em JavaScript, gerenciar o estado com previsibilidade está se tornando cada vez mais importante. A imutabilidade pode

Linguagens & Frameworks

Não consideramos mais Redux a abordagem padrão para gerenciamento de estado em aplicações React. Ainda é um framework valioso, mas pode levar a um código mais detalhado e mais difícil de seguir do que outras alternativas.

(Redux)

XState é um framework JavaScript e TypeScript simples, para criar máquinas de estado finito e visualizá-las como gráficos de estado.

(XState)

ajudar a garantir que nossas aplicações se comportem de forma consistente, mas infelizmente o JavaScript não oferece estruturas de dados profundamente imutáveis integradas (veja a proposta ES para Records e Tuples). Immer — que significa "sempre" em alemão — é um pequeno pacote que permite trabalhar com o estado imutável de uma forma mais conveniente. É baseado no mecanismo copy-on-write, tem uma API mínima e opera em objects e arrays JavaScript normais. Isso significa que o acesso aos dados é feito normalmente e nenhum grande esforço de refatoração é necessário ao introduzir a imutabilidade em uma base de código existente. Muitos de nossos times agora o usam em suas bases de código JavaScript e o preferem em relação ao Immutable.js, e é por isso que nós estamos o movendo para Experimente.

Redux
Experimente

Decidimos mover o Redux de volta ao anel Experimente para ilustrar que não o consideramos mais a abordagem padrão para gerenciamento de estado em aplicações React. Nossa experiência mostra que o Redux ainda é um framework valioso em muitos casos, mas comparado a outras abordagens, ele leva a um código mais detalhado e difícil de seguir. A inclusão do Redux Sagas geralmente aumenta esse problema. Como alternativa, você em geral pode usar os recursos das versões recentes do React para gerenciar o estado de maneira eficaz sem um framework adicional. No entanto, queremos destacar que quando você atinge o ponto em que sua solução de gerenciamento de estado simples começa a se tornar complexa, pode valer a pena recorrer ao Redux ou talvez ao recém-publicado pelo Facebook, Recoil.

Rust
Experimente

A linguagem de programação Rust continua a crescer em popularidade e foi eleita a linguagem "mais amada" do Stack Overflow por pessoas desenvolvedoras por cinco anos consecutivos. Nós também gostamos. É uma linguagem rápida, segura e expressiva que tem cada vez mais utilidade à medida que seu ecossistema cresce. Por exemplo, Rust está começando a ser usada para ciência de dados e aprendizado de máquina e pode fornecer um aumento significativo de desempenho. Há também o Materialize, um banco de dados de baixa latência orientado para streaming que é escrito em Rust.

single-spa
Experimente

single-spa é um framework JavaScript que reúne vários micro frontends em uma única aplicação frontend. Apesar de desaconselharmos a anarquia de micro frontends, ou seja, o uso de micro frontends como uma desculpa para combinar vários frameworks, o single-spa suporta exatamente isso. Entendemos que existem cenários legítimos, como atualizar para uma nova revisão de um framework em vários micro frontends, em que a integração entre vários frameworks é necessária. single-spa tem sido o framework de escolha para integração de micro frontends para nossos times. Eles também têm achado que o single-spa funciona bem com SystemJS e gerenciando diferentes versões de um dependência única.

Strikt
Experimente

O ecossistema Kotlin continua crescendo, e mais bibliotecas vêm se beneficiando dos recursos da linguagem Kotlin para substituição de alternativas Java. Strikt é uma biblioteca que permite escrever asserções de teste em um estilo muito fluente. Ela usa lambdas e blocos Kotlin para tornar seus testes menos prolixos, mantendo a legibilidade. Strikt também suporta a construção de asserções personalizadas, o que pode tornar seus testes mais específicos para o domínio.

XState
Experimente

Já apresentamos várias bibliotecas de gerenciamento de estado no Radar, mas XState tem uma abordagem um pouco diferente. É um framework JavaScript e TypeScript simples, para criar máquinas de estado finito e visualizá-las como gráficos de estado. Ele se integra com os frameworks JavaScript reativos mais populares (Vue.js, Ember.js, React.js e RxJS), e é baseado no padrão W3C para máquinas de estado finito. Outro recurso notável é a serialização de definições de máquina. Uma coisa que achamos útil ao criar máquinas de estados finitos em outros contextos (particularmente ao escrever

lógica de jogo) é a capacidade de visualizar estados e suas possíveis transições. Gostamos do fato de ser muito fácil fazer isso com o visualizador do XState.

Babylon.js
Avalie

Quando escrevemos sobre Realidade Virtual além dos jogos, há alguns anos, não fizemos nenhuma previsão sobre quão rapidamente e em que medida as soluções de VR seriam encontrados em áreas além de video games. Em retrospectiva, certamente vimos o interesse e a adoção crescerem, mas a aceitação foi mais lenta do que parte de nós imaginou. Um motivo pode ser o ferramental. Unity e Unreal são duas ferramentas bastante maduras e aptas para desenvolver aplicações de VR. Também destacamos Godot. No entanto, essas ferramentas são bem diferentes de outras com as quais a maioria dos times de desenvolvimento web e corporativo está acostumada. Conforme continuamos a explorar, notamos que as soluções de VR baseadas na web já evoluíram significativamente, e tivemos uma experiência positiva com Babylon.js. Escrito em TypeScript e renderizando suas aplicações no navegador, Babylon.js fornece uma experiência familiar para muitos times de desenvolvimento. Além disso, Babylon.js é um software de código aberto, maduro e bem fundamentado, o que o torna ainda mais atraente.

Blazor
Avalie

Embora o JavaScript e seu ecossistema sejam dominantes no espaço de desenvolvimento de UI para web, novas oportunidades estão se abrindo com o surgimento de WebAssembly. Vemos Blazor como uma opção interessante para construir UIs web interativas usando C#. Gostamos particularmente desse framework de código aberto pelo fato de permitir a execução de código C# no navegador em cima do WebAssembly, aproveitando o tempo de execução e o ecossistema do .NET Standard, bem como bibliotecas personalizadas desenvolvidas na linguagem. Além disso, ele pode interoperar de forma bidirecional com código JavaScript no navegador, se necessário.

Flutter Driver
Avalie

Flutter Driver é uma biblioteca de testes de integração para aplicações Flutter. Com Flutter Driver, você pode instrumentar e conduzir o conjunto de testes em dispositivos reais ou emuladores. Nossos times continuam a escrever testes de unidade e testes de widget para garantir que a maior parte da funcionalidade de negócio em aplicações Flutter seja implementada. No entanto, para testar a real interação do usuário, estamos avaliando o desempenho do Flutter Driver, e você também deveria.

HashiCorp Sentinel
Avalie

Apesar de defendermos fortemente a definição de políticas de segurança como código, o conjunto de ferramentas neste espaço tem se mantido bastante limitado. Se você estiver usando produtos HashiCorp (como Terraform ou Vault) e não se importar em pagar pelas versões corporativas, você tem a opção de usar o HashiCorp Sentinel. Sentinel é, na verdade, uma linguagem de programação completa para definir e implementar decisões de políticas baseadas em contexto. Por exemplo, no Terraform, ela pode ser usada para testar violações de políticas antes de aplicar alterações de infraestrutura. No Vault, pode-se usar Sentinel para definir controle de acesso granular nas APIs. Essa abordagem tem todos os benefícios de encapsulamento, manutenção, legibilidade e extensibilidade que as linguagens de programação de alto nível oferecem, criando uma alternativa atraente para as políticas de segurança declarativas tradicionais. Sentinel está na mesma classe de ferramentas do Open Policy Agent, mas é proprietária, tem código-fonte fechado e só funciona com produtos HashiCorp.

Hermes
Avalie

Hermes é uma ferramenta JavaScript otimizada para inicialização rápida de aplicações React Native em Android. Ferramentas de JavaScript como V8 possuem compiladores just-in-time (JIT) que compilam o código em tempo de execução para produzir instruções otimizadas. Hermes, no entanto, adota uma abordagem diferente, compilando o código JavaScript ahead of time (AOT) em um bytecode otimizado. Como resultado, você obtém um tamanho de imagem APK reduzido, menor consumo de memória e tempo de inicialização mais rápido. Estamos avaliando cuidadosamente o uso de Hermes em algumas aplicações React Native e recomendamos que você faça o mesmo.

Linguagens & Frameworks

Sentinel é uma linguagem de programação completa para definir e implementar decisões de políticas baseadas em contexto.

(HashiCorp Sentinel)

Linguagens & Frameworks

Kedro é um framework de fluxo de trabalho de desenvolvimento para projetos de ciência de dados, que traz uma abordagem padronizada para a construção de dados prontos para produção e pipelines de aprendizado de máquina.

(Kedro)

io-ts
Avalie

Temos gostado muito de usar o TypeScript há algum tempo, e adoramos a segurança que a tipagem forte oferece. No entanto, colocar dados dentro dos limites do sistema de tipos, por exemplo, de uma chamada para um serviço de back-end, pode levar a erros de tempo de execução. Uma biblioteca que ajuda a resolver esse problema é io-ts. Ela preenche a lacuna entre a verificação de tipo em tempo de compilação e o consumo em tempo de execução de dados externos, fornecendo funções de codificação e decodificação. Também pode ser usada como uma proteção de tipos personalizados. De acordo com nossos times, é uma solução elegante para um problema inconveniente.

Kedro
Avalie

No passado, falamos sobre o aprimoramento de ferramentas para aplicação de boas práticas de engenharia em projetos de ciência de dados. Kedro é outra boa adição neste espaço. É um framework de fluxo de desenvolvimento para projetos de ciência de dados que traz uma abordagem padronizada para a construção de dados prontos para produção e pipelines de aprendizado de máquina. Gostamos do foco nas práticas de engenharia de software e do bom design, com ênfase no desenvolvimento orientado a testes, modularidade, controle de versão e boas práticas de higienização, como manter as credenciais fora da base de código.

LitElement
Avalie

Um progresso constante tem acontecido desde que escrevemos sobre componentes web em 2014. LitElement, parte do Polymer Project, é uma biblioteca simples que pode ser usada para criar componentes web leves. Ela é realmente apenas uma classe base que elimina a necessidade de muito dos boilerplates comuns, tornando a programação de componentes web muito mais fácil. Tivemos sucesso ao usá-lo nas primeiras experiências em projetos e estamos otimistas para ver o amadurecimento da tecnologia.

Mock Service Worker
Avalie

Aplicações web, especialmente aquelas desenvolvidas para uso interno em empresas, geralmente são escritas em duas partes. A interface de usuário e parte da lógica de negócio são executadas no navegador web, enquanto a maior parte da lógica de negócio, autorizações e persistência são executadas em um servidor. Essas duas metades normalmente se comunicam via JSON por HTTP. Os endpoints não devem ser confundidos com uma API real; eles são simplesmente um detalhe de implementação de uma aplicação dividida em dois ambientes de tempo de execução. Ao mesmo tempo, fornecem um ponto de extensão válido para testar as peças individualmente. Ao testar a parte do JavaScript, o lado do servidor pode ser fragmentado e simulado no nível da rede por uma ferramenta como Mountebank. Uma abordagem alternativa é interceptar as solicitações no navegador. Gostamos da abordagem adotada pelo Mock Service Worker porque, com os service workers, a ferramenta usa uma abstração familiar para pessoas desenvolvedoras. Essa abordagem resulta em uma configuração mais simples e execução de teste mais rápida. No entanto, como eles não testam a camada de rede real, você deve implementar alguns testes de ponta a ponta como parte de uma pirâmide de testes íntegra.

Recoil
Avalie

Mais e mais times que usam React estão reavaliando suas opções de gerenciamento de estado, algo que também mencionamos em nossa reavaliação do Redux. Agora, o Facebook — que criou o React — publicou o Recoil, um novo framework para gerenciamento de estado, que surgiu a partir de uma aplicação interna que tinha que lidar com grandes quantidades de dados. Embora atualmente não tenhamos muita experiência prática com Recoil, vemos seu potencial. A API é simples e fácil de aprender, parece um React idiomático.

Ao contrário de outras abordagens, o Recoil fornece uma maneira eficiente e flexível de ter o estado compartilhado em uma aplicação: ele oferece suporte a estados criados dinamicamente por consultas e dados derivados, bem como a observação de estado em toda a aplicação sem prejudicar a divisão de código.

Snorkel
Avalie

Os modelos modernos de aprendizado de máquina são muito complexos e exigem grandes quantidades de conjuntos de dados de treinamento rotulados para aprender. Snorkel surgiu no laboratório de Inteligência Artificial da Stanford com a constatação de que rotular manualmente os dados é muito caro e muitas vezes inviável. O Snorkel nos permite rotular dados de treinamento de forma programática por meio da criação de funções de rotulagem. Ele emprega técnicas de aprendizado supervisionado para avaliar a precisão e as correlações das funções de rotulagem e, em seguida, determina seus pesos novamente e combina seus rótulos de saída, resultando em rótulos de treinamento de alta qualidade. O time que criou o Snorkel lançou posteriormente uma plataforma comercial chamada Snorkel Flow. Embora o Snorkel não seja mais ativamente desenvolvido, ele ainda é significativo por suas ideias sobre o uso de métodos pouco supervisionados para rotular dados.

Streamlit
Avalie

Streamlit é um framework de aplicação de código aberto em Python, usado por cientistas de dados para construir aplicações de visualização de dados com visuais bonitos. Streamlit se destaca em relação a concorrentes como Dash por seu foco em prototipagem rápida e suporte para uma ampla gama de bibliotecas de visualização, incluindo Plotly e Bokeh. Para cientistas de dados que precisam de showcases rápidos durante o ciclo de experimentação, Streamlit é uma escolha sólida. Estamos usando em alguns projetos e gostamos da possibilidade de criar visualizações interativas com muito pouco esforço.

Svelte
Avalie

Continuamos a observar novos frameworks JavaScript de front-end, e Svelte se destaca como um novo e promissor framework de componente. Ao contrário de outros frameworks que utilizam virtual DOM, Svelte compila em código JavaScript puro sem framework que atualiza diretamente o DOM de maneira cirúrgica. No entanto, é apenas um framework de componente; se você está planejando construir aplicações ricas em funcionalidades, avalie o uso do Sapper junto com o Svelte.

SWR
Avalie

SWR é uma biblioteca React Hooks para busca de dados remotos. Ela implementa a estratégia de cache HTTP stale-while-revalidate. A SWR primeiramente retorna os dados de cache (stale), depois envia a solicitação de busca (revalidate) e, finalmente, atualiza os valores com a resposta atualizada. Os componentes recebem um fluxo de dados, primeiro obsoletos e, em seguida, novos, de maneira constante e automática.

Nossos times de desenvolvimento tiveram boas experiências com a SWR, melhorando significativamente a experiência de uso por sempre ter dados na tela. No entanto, alertamos os times para que usem a estratégia de cache SWR apenas quando apropriada para uma aplicação retornar dados obsoletos. Observe que o HTTP requer que os caches respondam a uma solicitação com a resposta mais atualizada que seja apropriada para a solicitação, e apenas em *circunstâncias cuidadosamente avaliadas* uma resposta obsoleta pode ser retornada.

Testing Library
Avalie

Testing Library é uma família de pacotes para testar aplicações em vários frameworks, como React, Vue, React Native e Angular, entre outros. O conjunto de bibliotecas ajuda a testar os componentes da UI de uma forma centrada no usuário, incentivando você a testar o comportamento do usuário em vez de detalhes de implementação, como a presença de elementos na UI em um determinado momento. Entre os benefícios dessa mentalidade estão testes mais confiáveis, e é isso que destacamos como seu principal diferencial. Recomendamos que você avalie esta família de bibliotecas ao testar suas aplicações web em qualquer framework. Embora nossa experiência direta seja limitada a React Testing Library e Angular Testing Library, tivemos uma impressão positiva do que vimos.

Linguagens & Frameworks

Criado na Stanford University, o Snorkel nos permite rotular programaticamente os enormes conjuntos de dados usados para treinar algoritmos de aprendizado de máquina.

(Snorkel)

ThoughtWorks®

Somos uma consultoria global de software e uma comunidade de indivíduos apaixonados e guiados por propósitos, com mais de 7.000 pessoas em 43 escritórios distribuídos em 14 países. Em nossos mais de 25 anos de história, ajudamos clientes a resolver problemas de negócio complexos, usando a tecnologia como diferencial. Quando a mudança é a única constante, nós preparamos você para o imprevisível.

Quer se atualizar com artigos e informações relacionadas ao Radar?

Siga nossos perfis nas redes sociais e aproveite para se tornar assinante.

assine

ThoughtWorks®

thoughtworks.com/radar

#TWTechRadar