ThoughtWorks®

TECHNOLOGY RADAR VOL. 21

Um guia com opiniões firmes sobre as fronteiras da tecnologia

thoughtworks.com/radar
#TWTechRadar

CONTRIBUIÇÕES

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

Esta edição do Technology Radar da ThoughtWorks teve como base um encontro do Conselho Consultivo de Tecnologia, que se reuniu em São Francisco em outubro de 2019


Rebecca Parsons
(Diretora de Tecnologia)


Martin Fowler
(Cientista-chefe)


Bharani Subramaniam


Erik Dörnenburg


Evan Bottcher


Fausto de la Torre


Hao Xu


Ian Cartwright


James Lewis


Jonny LeRoy


Ketan Padegaonkar


Lakshminarasimhan Sudarshan


Marco Valtas


Mike Mason


Neal Ford


Ni Wang


Rachel Laycock


Scott Shaw


Shangqi Liu


Zhamak Dehghani

Tradução: Alexey Villas Boas, Daniela Araujo, Glauco Oliveira, Hellen Guareschi, Marco Valtas, Paula Ribas, Renan Martins, Ricardo Cavalcanti

SOBRE O RADAR

ThoughtWorkers são pessoas apaixonadas por tecnologia. Nós desenvolvemos, pesquisamos, testamos, contribuímos com código livre, escrevemos 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 para a 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 CTOs a pessoas que desenvolvem. O conteúdo é concebido para ser um resumo conciso.

Nós encorajamos você a explorar essas tecnologias para obter mais detalhes. O Radar é gráfico por natureza, agrupando os itens em técnicas, ferramentas, plataformas e linguagens & frameworks. Quando itens do radar puderem 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 AT A GLANCE

1 ADOTE

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

2 EXPERIMENTE

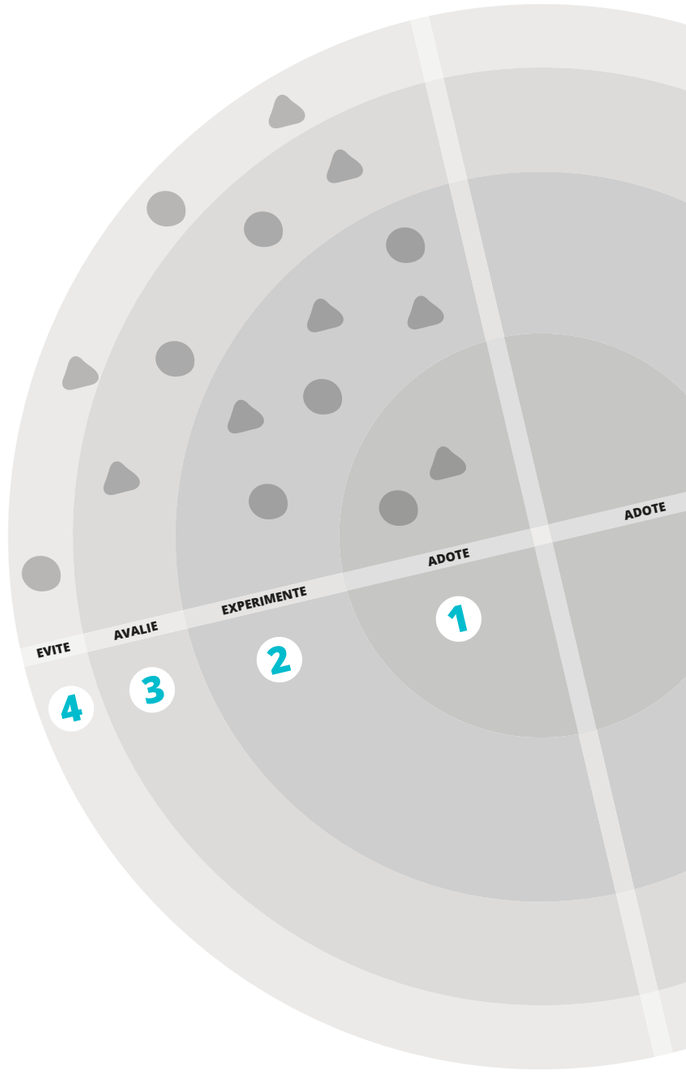
Vale a pena ir atrás. É importante entender como desenvolver essa capacidade. As empresas devem experimentar esta tecnologia em um projeto que possa lidar com o risco.

3 AVALIE

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

4 EVITE

Prossiga com cautela.



NOVO OU MODIFICADO
SEM MODIFICAÇÃO

Itens novos ou que sofreram alterações significativas desde o último Radar são representados como triângulos, enquanto os itens que não mudaram são representados como círculos.

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.

O QUE HÁ DE NOVO?

Temas em destaque nessa edição

Nuvem: mais é menos?

À medida que os maiores provedores de nuvem alcançaram quase a igualdade em sua funcionalidade central, o foco competitivo mudou para os serviços extras que podem fornecer, encorajando-os a lançar novas ofertas em uma velocidade vertiginosa. Em sua urgência para competir, lançam novos serviços com pontas soltas e funcionalidades incompletas. A ênfase na velocidade e a proliferação de produtos, por meio tanto da aquisição como de serviços precipitadamente criados, muitas vezes resulta não só em bugs como também em documentação ruim, automatização difícil e integração incompleta entre as próprias partes dos fornecedores. Isso causa frustração para os times que tentam entregar software com a funcionalidade prometida pelo provedor de nuvem e se deparam com constantes obstáculos. As empresas escolhem fornecedores de nuvem por uma variedade de fatores e, muitas vezes, em um nível alto na organização. Nosso conselho para os times: não suponham que todos os serviços de seu provedor de nuvem designado sejam iguais em qualidade, testem suas capacidades principais e estejam abertos para opções alternativas de código aberto ou uma estratégia de polycloud, caso seus próprios trade-offs para comercialização justifiquem a sobrecarga operacional de gerenciá-los.

Protegendo a cadeia de suprimento de software

As organizações devem resistir a regras de governança supervalorizadas que precisam de inspeção e aprovação manuais demoradas. Em vez disso, proteção de dependência automatizada (Função de aptidão de desvio de dependência), segurança (Política de segurança como código) e outros mecanismos de governança (Custo de execução como função de aptidão de arquitetura) protegem as partes do software *importantes*, mas não *urgentes*. Esse tópico que trata de políticas, compliance e governança como código reapareceu muitas vezes em nossas conversas. Vemos uma evolução natural no ecossistema de desenvolvimento de software no que diz respeito à ampliação da automação: integração contínua com testes automatizados, entrega contínua, infraestrutura como código e, agora, governança automatizada. Construir automação em torno do custo com nuvem, gerenciamento de dependência, estruturas de arquitetura e outros processos manuais antigos mostram uma evolução natural; estamos aprendendo como podemos automatizar todos os aspectos importantes da entrega de software.

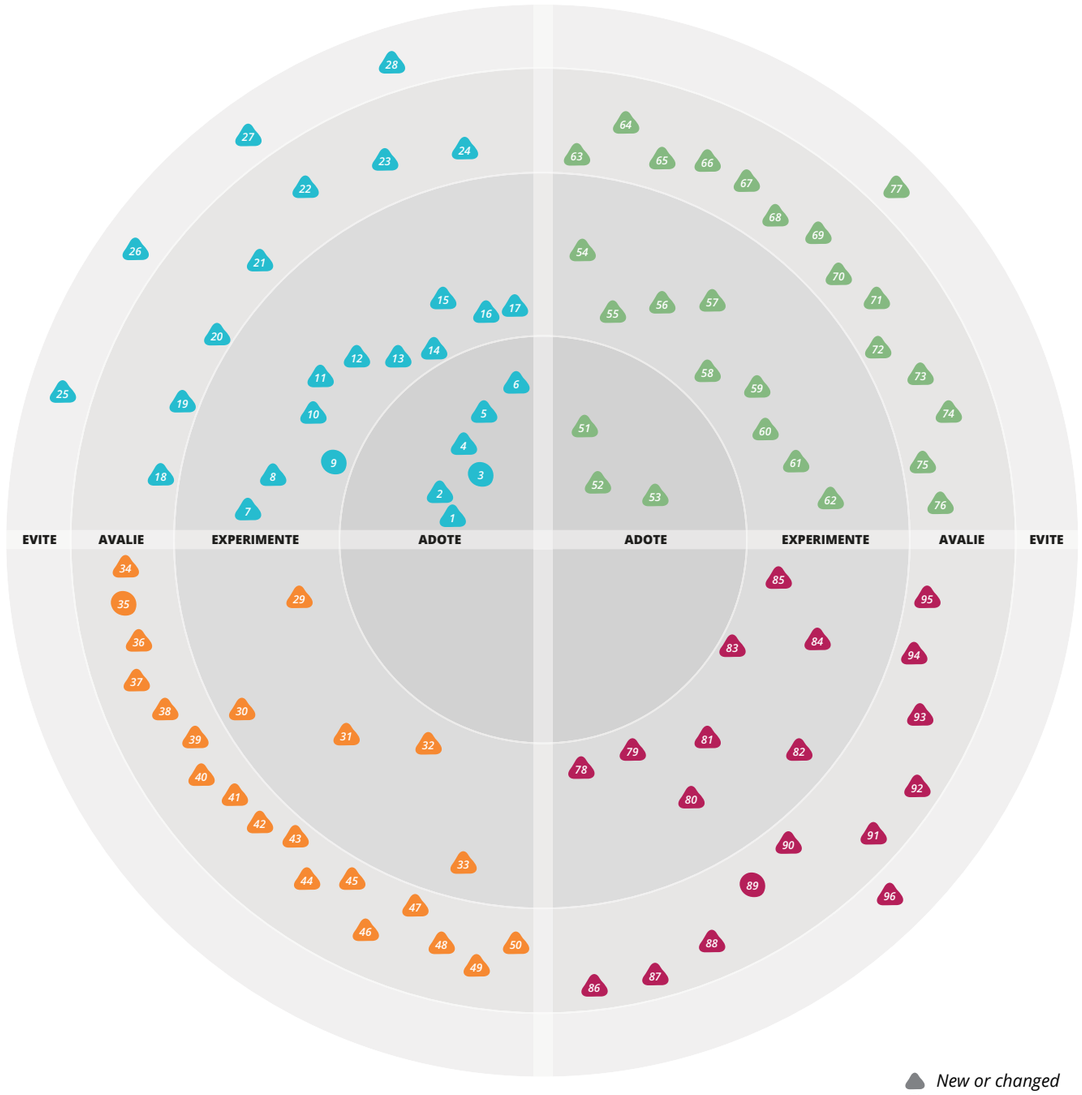
Interpretando a caixa preta do aprendizado de máquina

O aprendizado de máquina frequentemente parece descobrir soluções para os problemas que os humanos não conseguem, usando identificação de padrões, *back propagation* e outras técnicas bem conhecidas. Contudo, apesar de seu poder, muitos desses modelos são inerentemente obscuros, ou seja, seus resultados não podem ser explicados com inferência lógica. Isso é um problema na medida que humanos têm o direito de saber como uma decisão foi tomada ou quando há risco de introduzir vieses de preconceito, amostragem, algoritmos ou outros no modelo. Estamos vendo agora o surgimento de ferramentas como What-If e técnicas como o teste com viés ético que nos ajudam a encontrar as limitações e prever o resultado desses modelos. Embora essas melhoras em *interpretação* sejam um passo na direção certa, *explicar* redes neurais profundas ainda é um objetivo difícil de se alcançar. Por isso, cientistas de dados estão começando a considerar a explicabilidade como uma preocupação de primeira classe ao escolherem um modelo de aprendizado de máquina.

Desenvolvimento de software como um esporte de equipe

Desde o começo do nosso Technology Radar, advertimos sobre ferramentas e técnicas que isolam membros dos times de tecnologia uns dos outros, dificultando o feedback e a colaboração. Frequentemente, quando novas especializações aparecem, profissionais, fornecedores e ferramentas insistem que alguma parte do desenvolvimento deve ser feita em um ambiente isolado, longe do caos do ambiente "normal". Rejeitamos essa afirmação e constantemente procuramos por novas maneiras de retomar o desenvolvimento de software como um esporte de equipe. O feedback é muito importante quando desenvolvemos coisas complexas como software. Embora projetos cada vez mais precisem de especialização, lutamos para encaixá-los em colaboração e feedback regulares. Particularmente, não gostamos do meme "10x engineers" e preferimos focar na criação e na viabilização do "equipes 10x". Vemos isso atualmente influenciando a forma como design, ciência de dados e segurança podem ser integrados com times multifuncionais e apoiados em sólida automação. A próxima fronteira está trazendo mais atividades de governança e compliance para o jogo.

O RADAR



TÉCNICAS

ADOTE
1. Escaneamento de segurança de contêiner
2. Integridade de dados na origem
3. Micro-frontends
4. Pipelines para infraestrutura como código
5. Custo de execução como função de aptidão arquitetural
6. Testar usando dispositivos reais

EXPERIMENTE
7. Aprendizado de máquina automatizado (AutoML)
8. Atestado binário
9. Entrega contínua para aprendizado de máquina
10. Detecção de dados
11. Função de aptidão para controle de dependências
12. Sistemas de design
13. Ferramentas de rastreamento de experimentos para aprendizado de máquina
14. Explicabilidade como critério de seleção de modelo de primeira classe
15. Políticas de segurança como código
16. Sidecars para segurança de terminal
17. Zhong Tai

AVALIE
18. BERT
19. Malha de dados
20. Testes de viés ético
21. Aprendizado federado
22. JAMstack
23. Pareamento de registro para preservação de privacidade (PPRL) usando o filtro Bloom
24. Loops de aprendizado semissupervisionados

EVITE
25. Pessoas engenheiras 10x
26. Integração de front-end por meio de artefato
27. Lambda pinball
28. Paridade de funcionalidades em migração de legados

PLATAFORMAS

ADOTE

EXPERIMENTE
29. Apache Flink
30. Apollo Auto
31. GCP Pub/Sub
32. Mongoose OS
33. ROS

AVALIE
34. Kit de Desenvolvimento em Nuvem AWS
35. Azure DevOps
36. Pipelines Azure
37. Crowdin
38. Crux
39. Delta Lake
40. Fission
41. FoundationDB
42. GraalVM
43. Hydra
44. Kuma
45. MicroK8s
46. Oculus Quest
47. ONNX
48. Contêineres sem raiz
49. Snowflake
50. Teleport

EVITE

FERRAMENTAS

ADOTE
51. Commitizen
52. ESLint
53. Guiador de Estilo React

EXPERIMENTE
54. Bitrise
55. Dependabot
56. Detekt
57. Figma
58. Jib
59. Loki
60. Trivy
61. Twistlock
62. Yocto Project

AVALIE
63. Aplas
64. asdf-vm
65. AWSume
66. dbt
67. Docker Notary
68. Facets
69. Falco
70. in-toto
71. Kubeflow
72. MemGuard
73. Open Policy Agent (OPA)
74. Pumba
75. Skaffold
76. What-If Tool

EVITE
77. Azure Data Factory para orquestração

LINGUAGENS & FRAMEWORKS

ADOTE

EXPERIMENTE
78. Arrow
79. Flutter
80. jest-when
81. Micronaut
82. React Hooks
83. Biblioteca de Testes para React
84. Componentes estilizados
85. Tensorflow

AVALIE
86. Fairseq
87. Flair
88. Gatsby.js
89. GraphQL
90. KotlinTest
91. NestJS
92. Paged.js
93. Quarkus
94. SwiftUI
95. Testcontainers

EVITE
96. Enzyme

TÉCNICAS

Escaneamento de segurança de contêiner

ADOTE

A adoção contínua de contêineres para implantação, especialmente Docker, fez do escaneamento de segurança de contêiner uma técnica obrigatória, e movemos esta técnica para o anel Adote para refletir isso. Especificamente, os contêineres introduziram um novo caminho para questões de segurança, e é essencial que você use ferramentas para escanear e verificar contêineres durante os deploys. Preferimos usar ferramentas de escaneamento automatizadas, que rodam como parte da pipeline de implantação.

Integridade de dados na origem

ADOTE

Hoje em dia, a resposta de muitas organizações para destravar dados para uso analítico é construir um labirinto de pipelines de dados. Pipelines recuperam dados de uma ou múltiplas fontes, limpam e então os transformam e os movem para outro local para uso. Essa abordagem para gerenciamento de dados frequentemente deixa as pipelines de consumo com a difícil tarefa de verificar a integridade dos dados de entrada e construir uma lógica complexa para limpar os dados e atender o nível necessário de qualidade. O problema fundamental é que a fonte dos dados não tem incentivo e responsabilidade por fornecer dados de qualidade para seu público consumidor. Por isso, defendemos fortemente a integridade dos dados na origem , ou seja, qualquer fonte que forneça dados consumíveis deve descrever suas medidas de qualidade de dados explicitamente e garanti-las. A principal razão por trás disso é que os sistemas e times de origem são mais intimamente ligados com seus dados e mais bem posicionados para corrigir na fonte. A arquitetura de malha de dados vai um passo além, comparando dados consumíveis a um produto, onde a qualidade de dados e seus objetivos são atributos integrais de cada conjunto de dados compartilhado.

Micro-frontends

ADOTE

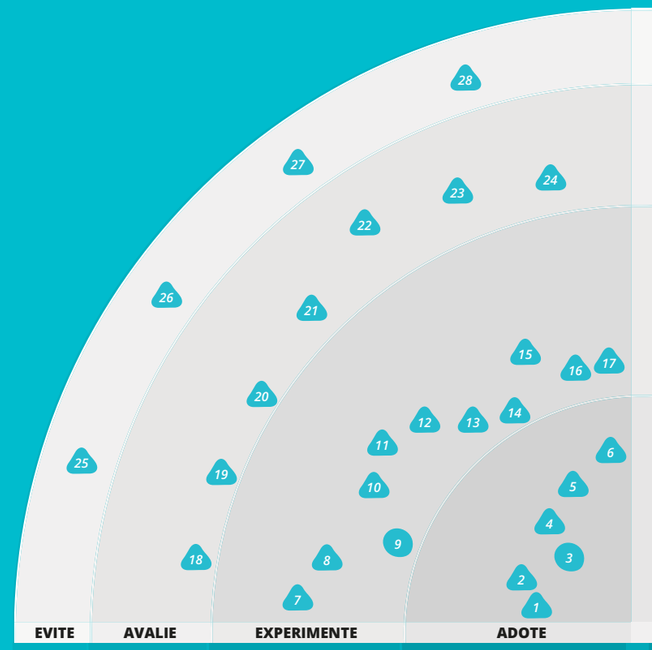
Temos visto significativos benefícios de se introduzir microsserviços, que permitiram que times escalassem a entrega de serviços mantidos e implantados independentemente. Infelizmente, também vimos muitos times criarem um monólito de front-end – uma grande aplicação de navegador confusa que fica em cima de serviços de back-end –, neutralizando fortemente os benefícios dos microsserviços. Os micro-frontends continuaram a ganhar popularidade desde que foram introduzidos. Temos visto muitos times adotarem alguma forma dessa arquitetura como uma maneira de gerenciar a complexidade de múltiplos desenvolvedores e times contribuindo para a mesma experiência do usuário. Em junho deste ano, um dos criadores dessa técnica publicou um artigo introdutório que serve como uma referência para micro-frontends. Ele mostra como esse estilo pode ser implementado usando-se vários mecanismos de programação web e constrói uma aplicação de exemplo usando

ADOTE
1. Escaneamento de segurança de contêiner
2. Integridade de dados na origem
3. Micro-frontends
4. Pipelines para infraestrutura como código
5. Custo de execução como função de aptidão arquitetural
6. Testar usando dispositivos reais

EXPERIMENTE
7. Aprendizado de máquina automatizado (AutoML)
8. Atestado binário
9. Entrega contínua para aprendizado de máquina
10. Detecção de dados
11. Função de aptidão para controle de dependências
12. Sistemas de design
13. Ferramentas de rastreamento de experimentos para aprendizado de máquina
14. Explicabilidade como critério de seleção de modelo de primeira classe
15. Políticas de segurança como código
16. Sidecars para segurança de terminal
17. Zhong Tai

AVALIE
18. BERT
19. Malha de dados
20. Testes de viés ético
21. Aprendizado federado
22. JAMstack
23. Pareamento de registro para preservação de privacidade (PPRL) usando o filtro Bloom
24. Loops de aprendizado semissupervisionados

EVITE
25. Pessoas engenheiras 10x
26. Integração de front-end por meio de artefato
27. Lambda pinball
28. Paridade de funcionalidades em migração de legados



Técnicas

Ferramentas de aprendizado de máquina automatizado (AutoML) surgiram para preencher a lacuna entre a oferta e a demanda de cientistas de dados especializados em aprendizado de máquina. Essas ferramentas são um ponto de partida útil – mas produzem melhores resultados quando usadas por especialistas.

(AutoML)

O atestado binário é uma técnica para implementar o controle de segurança no tempo de implantação, verificando criptograficamente se uma imagem binária está autorizada para implantação.

(Atestado binário)

o React.js. Estamos confiantes de que este estilo vai crescer em popularidade à medida que grandes organizações tentam dividir o desenvolvimento UI entre múltiplos times.

Pipelines para infraestrutura como código

ADOTE

O uso de pipelines de entrega contínua para orquestrar o processo de release para software se tornou um conceito comum. Ferramentas de CI/CD podem ser usadas para testar a configuração do servidor (ex.: cookbooks do Chef, módulos Puppet, playbooks do Ansible), construção de imagem do servidor (ex.: Packer), provisionamento de ambiente (ex.: Terraform, CloudFormation) e a integração de ambientes. O uso de pipelines para infraestrutura como código permite que você encontre erros antes das mudanças serem aplicadas em ambientes operacionais – incluindo ambientes usados para desenvolvimento e teste. Elas também oferecem uma maneira de assegurar que o ferramental de infraestrutura seja executado consistentemente, usando agentes de CI/CD em vez de estações de trabalho individuais. Nossos times têm tido bons resultado adotando essa técnica em seus projetos.

Custo de execução como função de aptidão arquitetural

ADOTE

Automatizar a 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.

Testar usando dispositivos reais

ADOTE

Ao adotarem a entrega contínua com sucesso, os times se esforçam para deixar os vários ambientes de teste o mais parecidos possível com o de produção. Isso os permite evitar um grupo de bugs que, caso contrário, apareceria apenas no ambiente de produção. Isso também vale para software desenvolvido para sistemas embarcados e Internet das Coisas. Se nós não executarmos nossos testes em ambientes realistas, podemos encontrar alguns bugs pela primeira vez somente em produção. Testar usando dispositivos reais ajuda a evitar esse problema ao assegurar que os dispositivos certos estão disponíveis na pipeline de entrega contínua.

Aprendizado de máquina automatizado (AutoML)

EXPERIMENTE

O poder e a promessa do aprendizado de máquina criaram uma demanda por expertise que ultrapassa a quantia de cientistas de dados que se especializam nessa área. Em resposta a essa lacuna de habilidades, temos visto o aparecimento de ferramentas de aprendizado de máquina automatizado (AutoML) que têm como objetivo tornar mais fácil para quem não é especialista automatizar o processo de ponta-a-ponta de seleção e treinamento do modelo. Exemplos incluem o AutoML do Google, o DataRobot, e a interface H2O AutoML. Embora tenhamos visto resultados promissores com essas ferramentas, aconselhamos as empresas a não vê-las como a soma total necessária em sua jornada em aprendizado de máquina. Como dito no site do H2O, "ainda há um bom pedaço de conhecimento e background em ciência de dados que é necessário para produzir modelos de aprendizado de máquina de alta performance". Confiar cegamente em técnicas automatizadas também aumenta o risco de introduzir vieses éticos ou tomar decisões que colocam minorias em desvantagem. Embora as empresas possam usar essas ferramentas como ponto de partida para gerar modelos úteis, treinados, encorajamos que elas procurem cientistas de dados com experiência para validar e refinar os resultados.

Atestado binário

EXPERIMENTE

À medida que o uso de contêineres, a implantação de grandes frotas de serviços por times autônomos e a velocidade

aumentada da entrega contínua vão se tornado uma prática comum para muitas organizações, surge a necessidade de controles de segurança de software automatizados para o momento da implantação. O atestado binário é uma técnica para implementar controle de segurança para a implantação, verificando criptograficamente que uma imagem binária está autorizada para implantação. Usando essa técnica, um atestador, um processo de compilação automatizado ou um time de segurança podem aprovar os binários que passaram nos testes e verificações de qualidade requeridos e estão autorizados para implantação. Serviços como o GCP Autorização Binária, ativado pelo Grafeas, e ferramentas com in-toto e Docker Notary suportam a criação de atestados e a validação de assinatura de imagens antes da implantação.

Entrega contínua para aprendizado de máquina

EXPERIMENTE

Com a popularidade crescente das aplicações baseadas em aprendizado de máquina e a complexidade técnica envolvida na sua construção, nossos times dependem fortemente de entrega contínua para aprendizado de máquina (CD4ML) para entregar tais aplicações com segurança, rapidez e de maneira sustentável. CD4ML é a disciplina de trazer princípios e práticas da entrega contínua para aplicações de aprendizado de máquina. Ela remove ciclos longos entre treinar modelos e colocá-los em produção. A CD4ML remove as entregas manuais entre diferentes times, pessoas engenheiras de dados, cientistas de dados e engenheiras de aprendizado de máquina no processo ponta-a-ponta de construir e implantar um modelo atendido por um aplicativo. Usando CD4ML, nossos times têm implementado com sucesso versionamento, testes e implantação automatizados de todos os componentes de aplicativos baseados em aprendizado de máquina: dados, modelo e código.

Detecção de dados

EXPERIMENTE

Um dos principais pontos de atrito para cientistas de dados e analistas em seu fluxo de trabalho é localizar os dados de que precisam, entendê-los e avaliar se são confiáveis para uso. Isso permanece um desafio devido aos metadados faltantes sobre as fontes de dados disponíveis e falta de funcionalidade adequada para pesquisar e localizar os dados. Encorajamos times que estão fornecendo conjuntos de dados analíticos ou construindo plataformas de dados a tornar a detecção de dados uma função de primeira classe de seus ambientes, para fornecer a habilidade de localizar facilmente dados disponíveis, detectar sua qualidade, entender sua estrutura e linhagem e ter acesso a eles. Tradicionalmente, essa função tem sido fornecida por soluções de catalogação de dados inchadas. Nos últimos anos, temos visto o crescimento de projetos de código aberto que estão melhorando a experiência de desenvolvimento tanto para fornecedores quanto para consumidores de dados para fazer uma coisa muito bem: tornar os dados detectáveis. O Amundsen, da Lyft, e o WhereHows, do Linkedin, estão entre essas ferramentas. O que gostaríamos de ver é uma mudança no comportamento dos fornecedores para compartilharem intencionalmente os metadados que ajudam na descoberta em favor de ferramentas de detecção que inferem informações de metadados parciais de silos de bases de dados de aplicações.

Função de aptidão para controle de dependências

EXPERIMENTE

Muitos times e organizações não têm uma maneira formal ou consistente de rastrear dependências técnicas em seu software. Esse problema frequentemente aparece quando esse software precisa ser alterado, momento em que o uso de uma versão desatualizada de uma biblioteca, API ou componente vai causar problemas ou atraso. A função de aptidão para controle de dependências é uma técnica para introduzir uma específica função de aptidão de arquitetura evolutiva para rastrear essas dependências com o tempo, dando assim um indicativo do possível trabalho necessário e se um problema em potencial está melhorando ou piorando.

Sistemas de design

EXPERIMENTE

À medida que o desenvolvimento de aplicativos se torna cada vez mais dinâmico e complexo, é um desafio alcançar a entrega eficaz de produtos acessíveis e utilizáveis que sejam consistentes em estilo. Os sistemas de design definem uma coleção de padrões de design, bibliotecas de componentes e bons designs e práticas de engenharia que asseguram essa consistência no desenvolvimento de produtos digitais. Achamos que sistemas de design são um acréscimo útil a nossa caixa de ferramentas quando trabalhamos com múltiplos times e disciplinas em desenvolvimento de produto, porque permite que times foquem em desafios mais estratégicos acerca do produto em si, sem a necessidade de reinventar a roda toda vez que precisam de um componente visual. Os tipos de

Técnicas

Redes neurais profundas têm demonstrado desempenho e precisão notáveis em uma gama ampla de problemas. Porém, à medida que o uso aumenta, também aumenta a importância de poder explicar como as decisões são tomadas.

(Explicabilidade como critério de seleção de modelo de primeira classe)

Técnicas

A complexidade do cenário de tecnologia atualmente exige o tratamento das políticas de segurança como código: definir e manter as políticas sob sistemas de controle de versão, validá-las automaticamente, implantá-las automaticamente e monitorar suas performances.

(Políticas de segurança como código)

Zhong Tai é uma abordagem para entregar modelos de negócios encapsulados. É projetada para ajudar uma nova classe de pequenos negócios a entregar serviços de primeira classe sem os custos de umas infraestrutura empresarial tradicional.

(Zhong Tai)

componentes e ferramentas que você usa para criar sistemas de design podem variar imensamente.

Ferramentas de rastreamento de experimentos para aprendizado de máquina

EXPERIMENTE

O trabalho diário em aprendizado de máquina frequentemente inclui uma série de experimentos na seleção de uma abordagem de modelagem, topologia de rede, treinamento de dados e várias otimizações ou ajustes no modelo. Pelo fato de muitos desses modelos serem ainda difíceis de interpretar ou explicar, cientistas de dados devem usar sua experiência e intuição para criar hipóteses de mudanças e então medir o impacto que essas mudanças têm no desempenho global do modelo. Conforme esses modelos se tornam cada vez mais comuns nos sistemas de negócios, várias ferramentas de rastreamento experimentais para aprendizado de máquina surgiram para ajudar investigadores a monitorar esses experimentos e trabalhar neles metodicamente. Embora não tenha surgido uma vencedora, ferramentas como MLflow ou Weights & Biases e plataformas como Comet ou Neptune introduziram rigor e repetibilidade em todo o fluxo de trabalho do aprendizado de máquina. Elas também facilitam a colaboração e ajudam a ciência de dados a se transformar de uma empreitada solitária em um esporte coletivo.

Explicabilidade como critério de seleção de modelo de primeira classe

EXPERIMENTE

Redes neurais profundas têm demonstrado desempenho e precisão notáveis em uma gama ampla de problemas. Dada a quantidade suficiente de dados de treinamento e uma topologia escolhida adequadamente, esses modelos atendem e excedem as capacidades humanas em certos espaços de problemas específicos. Contudo, eles são inerentemente obscuros. Embora partes de modelos possam ser reusadas por meio da transferência de aprendizado, raramente somos capazes de atribui significado inteligível por pessoas para estes elementos. Em contraste, um modelo explicável nos permite dizer como uma decisão foi tomada. Por exemplo, uma árvore de decisão produz uma cadeia de inferência que descreve o processo de classificação. A explicabilidade se torna crítica em certas indústrias reguladas ou quando nos preocupamos com o impacto ético de uma decisão. À medida que estes modelos são incorporados mais amplamente a importantes sistemas de negócios, é importante considerar a explicabilidade como critério de seleção de modelo de primeira classe. Apesar de seu poder, redes neurais podem não ser uma escolha adequada quando os requisitos de explicabilidade forem rigorosos.

Políticas de segurança como código

EXPERIMENTE

Políticas de segurança são regras e procedimentos que protegem nossos sistemas de ameaças e interrupções. Por exemplo, políticas de controle de acesso definem e fazem cumprir quem pode acessar quais serviços e recursos sob quais circunstâncias; já políticas de segurança de rede podem dinamicamente limitar a taxa de tráfego de um serviço específico. A complexidade do cenário de tecnologia atualmente exige o tratamento das políticas de segurança como código : definir e manter as políticas sob sistemas de controle de versão, validá-las automaticamente, implantá-las automaticamente e monitorar suas performances. Ferramentas como a Open Policy Agent, ou plataformas como Istio oferecem maneiras flexíveis de definição e execução de tais políticas e suportam a prática de políticas de segurança como código.

Sidecars para segurança de terminal

EXPERIMENTE

Muitas das soluções técnicas que construímos hoje rodam em ambientes cada vez mais complexos de polycloud ou nuvem híbrida com múltiplos componentes e serviços distribuídos. Sob essas circunstâncias, usamos dois princípios de segurança no começo de uma implementação: *rede com zero confiança*, ou seja, nunca confie na rede e sempre verifique; e o princípio do *mínimo privilégio*, dando o mínimo de permissões necessárias para realizar um trabalho em particular. Os sidecars para segurança de terminal são uma técnica comum que usamos para implementar esses princípios para aplicar controles de segurança em cada terminal de componente (ex.: APIs de serviços, armazéns de dados ou interface de controle Kubernetes). Fazemos isso usando um sidecar fora do processo – um processo ou um contêiner que é implantado ou agendado com cada serviço, compartilhando o mesmo contexto de execução, hospedagem e identidade. Open Policy Agent e Envoy são ferramentas que implementam essa técnica. Sidecars para segurança de terminal minimizam a área de cobertura confiável para um terminal local, em vez do perímetro de rede. Gostamos de ver a responsabilidade da configuração da política de segurança do sidecar com o time que é responsável pelo terminal e não um time centralizado separado.

Zhong Tai
EXPERIMENTE

Zhong Tai é um chavão na indústria de TI chinesa há anos, mas ainda não se popularizou no Ocidente. Em essência, Zhong Tai é uma abordagem para entregar modelos de negócios encapsulados. É projetada para ajudar uma nova classe de pequenos negócios a entregar serviços de primeira classe sem os custos de umas infraestrutura empresarial tradicional, permitindo que organizações existentes tragam serviços inovadores para o mercado em uma velocidade vertiginosa. A estratégia Zhong Tai foi originalmente proposta pelo Alibaba e logo foi seguida por muitas empresas de Internet chinesas, pois seu modelo de negócio é digitalmente nativo para se replicar para novos mercados e setores. Hoje em dia, mais empresas chinesas estão usando Zhong Tai como uma alavanca para a transformação digital.

BERT
AVALIE

BERT significa *Bidirectional Encoder Representations from Transformers*. É um novo método de pré-treino de representações de linguagem que foi publicado por pesquisadores do Google, em outubro de 2018. BERT alterou significativamente o panorama do processamento de linguagem natural (NLP, em inglês) ao obter resultados de ponta em NLP. Baseado na arquitetura Transformer, ele aprende com contexto do lado esquerdo e do lado direito de um token durante o treinamento. O Google também lançou modelos BERT de uso geral pré-treinados em um grande corpo de texto sem tags, incluindo a Wikipedia. Pessoas desenvolvedoras podem usar e ajustar esses modelos pré-treinados em seus dados para tarefas específicas e conseguir grandes resultados. Falamos sobre transferir aprendizado para NLP em nossa edição de abril de 2019 do Radar. O BERT e seus sucessores continuam a fazer da transferência de aprendizado para NLP uma área muito empolgante, com significativa redução do esforço para usuários lidando com classificação de texto.

Malha de dados
AVALIE

A malha de dados é um paradigma arquitetônico que destrava dados analíticos em escala, rapidamente desbloqueando o acesso a um número cada vez maior de conjuntos de dados de domínio distribuído, para uma proliferação de cenários de uso, tais como aprendizado de máquina, analytics ou aplicações com uso intensivo de dados em toda a organização. A malha de dados aborda os modos que comumente falham no tradicional e centralizado lago de dados ou na arquitetura de plataforma de dados, com uma mudança do paradigma centralizado de um lago ou seu predecessor, o armazém de dados. A malha de dados muda para um paradigma baseado na arquitetura moderna distribuída: considerar domínios como a principal preocupação, usando mentalidade de plataformas para criar uma infraestrutura de dados de autosserviço, tratando dados como um produto e implementando a padronização padrão para permitir um ecossistema de produtos de dados distribuídos interoperáveis.

Testes de viés ético
AVALIE

No último ano, temos visto uma mudança no interesse pelo aprendizado de máquina e redes neurais profundas, em particular. Até agora, o desenvolvimento de ferramentas e técnicas tem sido guiado pela empolgação gerada pelas capacidades notáveis desses modelos. Atualmente, contudo, há uma preocupação crescente de que esses modelos possam causar prejuízo não-intencional. Por exemplo, um modelo pode ser treinado para tomar decisões de crédito lucrativas simplesmente excluindo pessoas candidatas desfavorecidas. Felizmente, estamos vendo um interesse crescente em testes de viés ético , que ajudarão a descobrir decisões potencialmente prejudiciais. Ferramentas, tais como lime, AI Fairness 360 ou What-If podem ajudar a descobrir imprecisões que resultam de grupos sub-representados em dados de treinamento, enquanto ferramentas de visualização como Google Facets ou Facets Dive podem ser usadas para descobrir subgrupos dentro de um conjunto de dados de treinamento. Contudo, esse é um campo em desenvolvimento e esperamos que os padrões e práticas específicas para testes de viés ético surjam com o tempo.

Aprendizado federado
AVALIE

Treinamento de modelo geralmente requer coleta e transporte de dados de sua fonte para uma localização centralizada onde o algoritmo de treinamento é executado. Isso se torna particularmente problemático quando os dados em treinamento consistem de informações pessoalmente identificáveis. Estamos otimistas com o aparecimento do aprendizado federado como um método de treinamento que preserva a privacidade em um grande conjunto de dados relacionados a indivíduos. As técnicas de aprendizado federado permitem que os dados permaneçam no dispositivo do usuário, sob seu controle, e ainda contribuindo para um conjunto de dados de treinamento.

Técnicas

Estamos otimistas com o aparecimento do aprendizado federado como um método de treinamento que preserva a privacidade em um grande conjunto de dados relacionados a indivíduos.

(Aprendizado federado)

Técnicas

JAMstack pode fornecer experiências de uso ricas em aplicações web que dependem, na maioria das vezes, de APIs e ofertas SaaS.

(JAMstack)

Assim, cada dispositivo do usuário atualiza um modelo independentemente; então, os parâmetros do modelo, em vez dos dados em si, são combinados em uma visualização centralizada. Largura de banda e limitações computacionais do dispositivo apresentam desafios técnicos significativos, mas gostamos da maneira que o aprendizado federado deixa o usuário no controle de sua própria informação pessoal.

JAMstack

AVALIE

A tendência que começou como backend como serviço para aplicações mobile nativas muitos anos atrás está agora se tornando popular nas aplicações web. Estamos vendo frameworks como Gatsby.js, que combina geração de site estático com renderização do lado do cliente com APIs de terceiros. Chamado de JAMstack (o JAM significa: JavaScript, API, e Markup), essa abordagem pode fornecer experiências ricas para o usuário em aplicações web que dependem, na maioria das vezes, de APIs e ofertas SaaS. Pelo fato de o HTML ser renderizado tanto no browser de web ou no momento da compilação, o modelo de implantação é o mesmo de sites gerados de forma totalmente estática, com todos seus benefícios: a superfície de ataque no servidor é pequena e uma ótima performance pode ser alcançada com baixo uso de recursos. Essas implantações são também ideais para uma rede de entrega de conteúdo. Na verdade, brincamos com a ideia de rotular essa técnica como aplicações *CDN first.*

Pareamento de registro para preservação de privacidade (PPRL) usando o filtro Bloom

AVALIE

Parear registros de diferentes fornecedores de dados na presença de uma chave compartilhada é algo trivial. Contudo, nem sempre você vai ter uma chave compartilhada e, mesmo que tenha, pode não ser uma boa ideia expô-la por questões de privacidade. O pareamento de registros para preservação de privacidade usando o filtro Bloom (uma estrutura de dados probabilísticos com eficiência de espaço) é uma técnica estabelecida que permite pareamento probabilístico de registros a partir de diferentes provedores de dados sem expor dados pessoais privados identificáveis. Por exemplo, ao parear dados de dois provedores de dados, cada um vai encriptar seus dados pessoais identificáveis usando o filtro Bloom para obter chaves de registro criptográficas de cada provedor. Entre outras técnicas, achamos que o pareamento de registros para preservação da privacidade usando filtros Bloom é escalável para grandes conjuntos de dados.

Loops de aprendizado semissupervisionados

AVALIE

Loops de aprendizado semissupervisionados são uma classe de fluxos de trabalho de aprendizado de máquina iterativo que tiram vantagem das relações encontradas em dados não-rotulados. Essas técnicas podem melhorar modelos ao combinar conjuntos de dados rotulados e não-rotulados de várias maneiras. Em outros casos, eles comparam modelos treinados em diferentes subconjuntos de dados. Diferente tanto do aprendizado não-supervisionado, em que uma máquina infere classes em dados não-rotulados, ou técnicas supervisionadas em que o conjunto em treinamento é totalmente rotulado, as técnicas semissupervisionadas levam a vantagem de um pequeno conjunto de dados rotulados e um conjunto bem maior de dados não-rotulados. O aprendizado semissupervisionado também se aproxima de técnicas de aprendizado ativo, em que um humano é direcionado a rotular seletivamente pontos de dados ambíguos. Já que humanos especialistas que podem rotular precisamente dados são um recurso escasso, e rotulagem é frequentemente uma atividade que consome muito tempo no fluxo de trabalho no aprendizado de máquina, as técnicas semissupervisionadas baixam o custo do treinamento e tornam o aprendizado de máquina possível para uma nova classe de usuários.

Pessoas engenheiras 10x

EVITE

O velho termo pessoas engenheiras 10x esteve sob escrutínio nos últimos meses. Uma thread amplamente compartilhada no Twitter basicamente sugere que empresas perdoem comportamentos antissociais e prejudiciais para manter pessoas engenheiras que são percebidas como tendo imenso rendimento individual. Felizmente, muitas

pessoas nas mídias sociais fizeram piada com o conceito, mas o estereótipo de "pessoa desenvolvedora rockstar" ainda é universal. Em nossa experiência, grandes pessoas engenheiras são guiadas não apenas pelo rendimento individual, mas por trabalhar em times incríveis. É mais eficaz construir times de indivíduos talentosos com experiências mistas e históricos diversificados e fornecer os ingredientes certos para o trabalho em equipe, aprendizado e melhoria contínua. Estes times 10x podem se mover mais rápido, escalar mais rapidamente e são muito mais resilientes – sem precisar ceder a maus comportamentos.

Integração de front-end por meio de artefato

EVITE

Quando times adotam o conceito de micro-frontends, eles têm um número de padrões ao seu dispor para integrar os micro-frontends individuais em uma aplicação. Como sempre, há antipadrões também. Um comum neste caso é a integração de front-end por meio de artefato. Para cada micro-frontend um artefato é construído, normalmente um pacote NPM, que é colocado num registro. Um passo depois, às vezes em uma pipeline de build diferente, então combina os pacotes individuais em um pacote final que contém todos os micro-frontends. De uma perspectiva puramente técnica, essa integração na hora do build resulta em uma aplicação funcionando. Contudo, integrar via artefato implica que, a cada mudança, o artefato todo precisar ser reconstruído, o que consome tempo e provavelmente terá um impacto negativo na experiência da pessoa desenvolvedora. Ainda pior, este estilo de integrar front-ends também introduz dependências diretas entre os micro-frontends no momento da compilação, causando, dessa forma, uma considerável sobrecarga de coordenação.

Lambda pinball

EVITE

Temos construído arquiteturas sem servidor em nossos projetos há alguns anos, e percebemos que é bem fácil cair na armadilha de construir um monolito distribuído. As arquiteturas Lambda pinball caracteristicamente perdem de vista importantes lógicas de domínio na rede emaranhada de lambdas, buckets e filas à medida que os requisitos oscilam em gráficos cada vez mais complexos de serviços de nuvem. Normalmente, eles são difíceis de testar como unidades, e a aplicação precisa ser testada como um todo integrado. Um padrão que podemos usar para evitar essas arquiteturas pinball é fazer uma distinção entre interfaces públicas e publicadas e aplicar os bons e velhos limites de domínio com interfaces publicadas entre eles.

Paridade de funcionalidades em migração de legados

EVITE

Estamos descobrindo que mais e mais empresas precisam substituir sistemas legados envelhecidos para acompanhar as demandas de seus clientes (tanto internos quanto externos). Um antipadrão que continuamos vendo é a paridade de funcionalidades em migração de legados , o desejo de manter a paridade de funcionalidades com os sistemas antigos. Vemos isso como uma grande oportunidade perdida. Frequentemente, os sistemas antigos têm inchado com o tempo, com muitos recursos não usados pelos usuários (50%, segundo um relatório de 2014 do Standish Group) e processos de negócios que evoluíram com o passar do tempo. Nosso conselho: convença seus clientes a dar um passo atrás, entendendo o que seus usuários *precisam* atualmente e priorizando essas necessidades, de acordo com resultados de negócios e métricas – isso é muitas vezes mais fácil de falar do que de fazer. Isso significa conduzir uma pesquisa sobre o usuário e aplicar práticas modernas de desenvolvimento de produto em vez de simplesmente substituir as práticas existentes.

Técnicas

Em nossa experiência, grandes pessoas engenheiras são guiadas não apenas pelo rendimento individual, mas por trabalhar em times incríveis.

(Pessoas engenheiras 10x)

PLATAFORMAS

Apache Flink
EXPERIMENTE

O Apache Flink teve uma adoção crescente desde nossa avaliação inicial, em 2016. A plataforma é reconhecida como o mecanismo líder de processamento de fluxo e também amadureceu gradualmente nas áreas de processamento de lote e aprendizado de máquina. Um dos diferenciais-chave do Flink em relação a outros mecanismos de processamento de fluxo é seu uso de checkpoints consistentes do estado de uma aplicação. No caso de uma falha, a aplicação é reiniciada e seu estado é carregado a partir do último checkpoint – para que a aplicação possa continuar processando como se a falha nunca houvesse acontecido. Isso nos ajuda a reduzir a complexidade na construção e na operação de sistemas externos para tolerância a falhas. Estamos vendo mais e mais empresas usando Flink para construir sua própria plataforma de processamento de dados.

Apollo Auto
EXPERIMENTE

A tecnologia de direção autônoma costumava ser exclusiva das gigantes da tecnologia. Apollo Auto a trouxe para as empresas tradicionais de automóveis. O objetivo do programa Apollo, de propriedade da Baidu, é se tornar a plataforma Android para a indústria da direção autônoma. A plataforma Apollo tem componentes como percepção, simulação, planejamento e controle inteligente que possibilita que as empresas de carro integrem seus próprios sistemas de direção autônoma no hardware de seus veículos. A comunidade desenvolvedora ainda é nova, mas com muitos fornecedores se juntando a ela para contribuir com mais implementações. Um de nossos projetos ajudou nosso cliente a completar exames de licenciamento de direção autônoma com o sistema de piloto automático baseado em Apollo. A plataforma também fornece uma abordagem arquitetônica evolutiva para adoção de funcionalidades avançadas gradualmente, o que possibilita que integremos mais sensores e funções de maneira ágil e iterativa.

GCP Pub/Sub
EXPERIMENTE

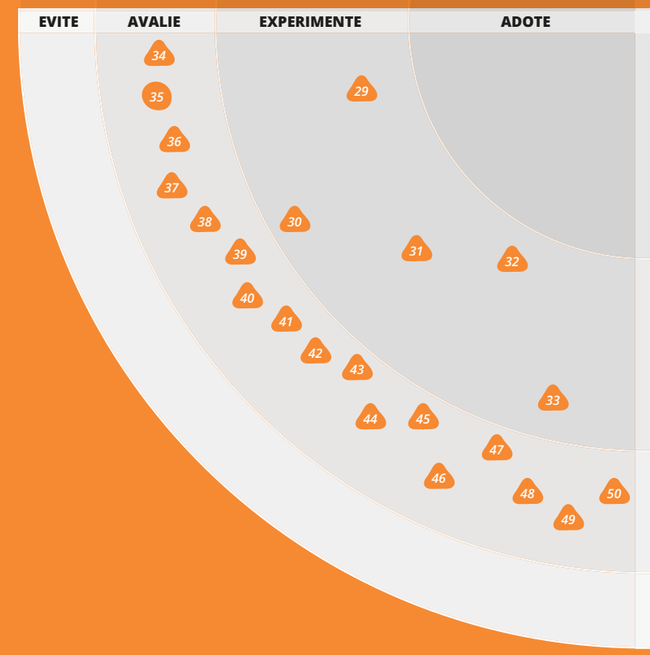
GCP Pub/Sub é a plataforma de streaming de eventos da Google Cloud. É uma peça de infraestrutura popular para muitas de nossas arquiteturas rodando a Plataforma Google Cloud, incluindo ingestão de eventos em massa, comunicação de cargas de trabalho sem servidor e fluxos de trabalho de processamento de dados em streaming. Uma de suas funcionalidades exclusivas é o suporte a inscrições pull e push: inscrição para receber todas as mensagens publicadas disponíveis na hora da inscrição ou envio de mensagens para

ADOTE

EXPERIMENTE
29. Apache Flink
30. Apollo Auto
31. GCP Pub/Sub
32. Mongoose OS
33. ROS

AVALIE
34. Kit de Desenvolvimento em Nuvem AWS
35. Azure DevOps
36. Pipelines Azure
37. Crowdin
38. Crux
39. Delta Lake
40. Fission
41. FoundationDB
42. GraalVM
43. Hydra
44. Kuma
45. MicroK8s
46. Oculus Quest
47. ONNX
48. Contêineres sem raiz
49. Snowflake
50. Teleport

EVITE



PLATAFORMAS

Apache Flink é o mecanismo líder de processamento de fluxo e também amadureceu gradualmente nas áreas de processamento de lote e aprendizado de máquina.

(Apache Flink)

O objetivo do programa Apollo, de propriedade da Baidu, é se tornar a plataforma Android para a indústria da direção autônoma.

(Apollo Auto)

um terminal em particular. Nossos times gostaram de sua confiabilidade e escala e do fato de funcionar como anunciado.

Mongoose OS

EXPERIMENTE

O Mongoose OS permanece um de nossos sistemas operacionais microcontroladores de código aberto e framework de desenvolvimento com firmware embarcado preferidos. É válido notar que o Mongoose OS preenche uma visível lacuna para pessoas desenvolvedoras de softwares embarcados: a lacuna entre o firmware Arduino adequado para prototipagem e SDKs nativos dos microcontroladores bare-metal. Nossos times têm usado com sucesso a mDash, nova plataforma para gerenciamento de dispositivo ponta-a-ponta da Cesanta's para projetos de hardware greenfield de pequena escala. Grandes fornecedores de plataformas em nuvem para Internet das Coisas hoje suportam o framework de desenvolvimento do Mongoose OS para gerenciamento de seus dispositivos, conectividade e atualizações de firmware over-the-air (OTA). Desde a última vez que falamos do Mongoose OS, o número de placas e microcontroladores cresceu e incluiu STM, Texas Instruments e Espressif. Continuamos a aproveitar seu suporte suave para atualizações OTA e sua segurança incorporada em nível de dispositivo individual.

ROS

EXPERIMENTE

ROS é um conjunto de bibliotecas e ferramentas para ajudar pessoas desenvolvedoras de software a criar aplicações robô. É um framework de desenvolvimento que fornece abstração de hardware, drivers de dispositivo, bibliotecas, visualizadores, message-passing, gerenciamento de pacotes e mais. A Apollo Auto é baseada em ROS. Em nosso outro projeto de simulação ADAS também usamos o sistema de mensagens do ROS (bag). A tecnologia não é nova, mas recuperou a atenção de pessoas desenvolvedoras com o desenvolvimento do ADAS.

Kit de Desenvolvimento em Nuvem AWS

AVALIE

Para muitos de nossos times, Terraform se tornou a escolha padrão para definir uma infraestrutura de nuvem. Contudo, alguns de nossos times têm experimentado o Kit de Desenvolvimento em Nuvem AWS e estão gostando do que viram até agora. Em particular, eles gostam do uso de linguagens de programação de primeira classe, em vez de arquivos de configuração, o que permite que usem ferramentas, abordagens de testes e habilidades existentes. Como ferramentas parecidas, ainda é preciso cuidado para assegurar que implantações permaneçam fáceis de entender e manter. Dado que o suporte para C# e Java vem em breve e ignorando, por enquanto, algumas lacunas na funcionalidade, achamos que o Kit AWS vale ser visto como uma alternativa a outras abordagens baseadas em arquivos de configuração.

Azure DevOps

AVALIE

Os serviços da Azure DevOps incluem um conjunto de serviços gerenciados, como os repositórios Git hospedados, pipelines de integração contínua/entrega contínua, ferramentas para testes automatizados, ferramenta para gerenciamento de backlog e repositório de artefatos. As Pipelines Azure DevOps têm amadurecido com o tempo. Nós particularmente gostamos de sua habilidade de definir Pipelines como código e seu ecossistema de extensões no marketplace da Azure DevOps. À época em que este texto foi escrito, nossos times ainda estavam encontrando algumas funcionalidades imaturas, incluindo a falta de uma UI eficaz para visualização e navegação de pipeline e a inabilidade de acionar uma pipeline a partir de artefatos ou outras pipelines.

Pipelines Azure

AVALIE

Pipelines Azure são um produto da Azure DevOps que oferecem soluções baseadas em nuvem para implementar pipelines como código em projetos hospedados no servidor Git Azure DevOps ou outra solução Git, como GitHub ou Bitbucket. A parte interessante dessa solução é a habilidade de executar seus scripts em Linux, MacOS e Windows sem a sobrecarga de gerenciar uma máquina virtual por si só. Isso representa um grande passo, especialmente para time que trabalham em ambientes Windows com soluções de frameworks .NET; também estamos avaliando esse serviço para entrega contínua em iOS.

Crowdin

AVALIE

A maioria dos projetos com suporte multi-idiomas começa com o time de desenvolvimento construindo funcionalidades em um idioma e gerenciando o resto por meio de traduções offline via e-mails e planilhas. Embora

esse esquema simples funcione, as coisas podem sair do rumo rapidamente. Você pode ter que responder repetidamente às mesmas questões para diferentes tradutores, esgotando a energia de colaboração entre tradutores, revisores e time desenvolvimento. Crowdin é uma das poucas plataformas que ajuda a simplificar o fluxo de trabalho de localização de seu projeto. Com Crowdin, o time de desenvolvimento pode continuar a construir funcionalidades e a plataforma organiza o texto que precisa de tradução em um fluxo de trabalho online. Gostamos do fato de Crowdin incentivar o time incorporar a tradução continuamente e incrementalmente, em vez de conduzi-la para grandes lotes no final do trabalho.

Crux
AVALIE

Crux é um banco de dados de documentos de código aberto com consultas bitemporais de grafos. A maioria dos sistemas de base de dados é temporal, ou seja, nos ajuda a modelar fatos na hora em que ocorrem. Sistemas de base de dados bitemporais permitem que você modele não apenas o tempo *válido* em que o fato ocorreu, mas também o tempo de *transação* de quando foi recebido. Se você precisa armazenar documentos com capacidades de grafos para consultar o conteúdo, dê uma chance ao Crux. Ele está atualmente em alfa e falta suporte SQL, mas você pode usar uma interface de consulta Datalog para ler e examinar relações.

Delta Lake
AVALIE

Delta Lake é uma camada de armazenamento de código aberto da Databrick que tenta trazer transações para processamento de big data. Um dos problemas que frequentemente encontramos quando usamos Apache Spark, é a falta de transações ACID. Delta Lake tem integração com API Spark e resolve esse problema usando um log de transação e arquivos Parquet versionados. Seu isolamento serializável permite que leitores e gravadores concorrentes operem em arquivos Parquet. Outras funcionalidades bem-vindas incluem validação de esquema na escrita e no versionamento, o que nos permite consultar e reverter para versões antigas de dados, se necessário. Começamos a usar em alguns de nossos projetos e temos gostado bastante.

Fission
AVALIE

O ecossistema sem servidor do Kubernetes está crescendo. Falamos sobre o Knative em uma edição anterior do Radar; agora estamos vendo o Fission ganhar força. O Fission permite que as pessoas desenvolvedoras se concentrem em escrever funções de curta duração e as mapeia para chamadas HTTP enquanto o framework lida com o resto das conexões e automação dos recursos Kubernetes por baixo dos panos. O Fission também permite a você compor funções, integrar com provedores terceirizados via retorno de chamada web (webhooks) e automatizar o gerenciamento da infraestrutura Kubernetes.

FoundationDB
AVALIE

O FoundationDB é um banco de dados multimodelos de código aberto, comprada pela Apple em 2015 e que se tornou código aberto em abril de 2018. O core do FoundationDB é um armazém distribuído de valor-chave, que fornece transações de serialização estritas. Um dos aspectos interessantes do FoundationDB é seu conceito de camadas para oferecer modelos adicionais. Essas camadas são essencialmente componentes stateless construídos em cima do armazém de valor-chave, como a camada Record e a Document layer. O FoundationDB define um padrão alto com seu teste Simulation, em que se rodam testes diários simulando várias falhas do sistema. Com sua performance, testes rigorosos e fácil operação, o FoundationDB não é apenas um banco de dados, podendo também ser usado por quem procura construir sistemas distribuídos em que podem usar o FoundationDB como um core primitivo sobre o qual se constrói seu sistema.

GraalVM
AVALIE

GraalVM é uma máquina virtual universal criada pela Oracle para executar aplicações escritas em linguagens JVM, JavaScript, Python, Ruby e R, assim como C/C++ e outras linguagens baseadas em LLVM. Na sua forma mais simples, GraalVM pode ser usada como uma máquina virtual com melhor desempenho para essas linguagens. Mas ela também nos permite escrever aplicações poliglotas com pouco impacto no desempenho, e sua utilidade de imagem nativa (atualmente disponível apenas como uma tecnologia early adopter) nos permite compilar o código Java antes do tempo para executáveis independentes, acarretando em uma inicialização mais rápida e menos uso de memória. A GraalVM gerou muita empolgação na comunidade Java e vários dos frameworks Java (incluindo Micronaut, Quarkus e Helidon) já estão tirando proveito dela.

PLATAFORMAS

GraalVM gerou muita empolgação na comunidade Java e vários dos frameworks Java (incluindo Micronaut, Quarkus e Helidon) já estão tirando proveito dela.

(GraalVM)

Kuma é uma malha de serviços agnóstica de plataforma para Kubernetes, VMs e ambientes bare metal.

(Kuma)

PLATAFORMAS

A interoperabilidade entre ferramentas e frameworks no ecossistema de redes neurais tem sido um desafio. ONNX pode ajudar.

(ONNX)

Hydra
AVALIE

Nem todo mundo precisa de uma solução OAuth2 auto-hospedada, mas, se este é o seu caso, consideramos Hydra — um servidor OAuth2 open source totalmente compatível e provedor de conexão OpenID — bem útil. Realmente gostamos do fato de que a Hydra não fornece nenhuma solução de gerenciamento de identidade out-of-the-box. Então, não importa qual tipo de gerenciamento de identidade você tem, é possível integrá-lo com a Hydra por meio de uma API limpa. Essa clara separação de identidade do resto do framework OAuth2 torna mais fácil integrar Hydra com um ecossistema de autenticação existente.

Kuma
AVALIE

Kuma é uma malha de serviços agnóstica de plataforma para Kubernetes, VMs e ambientes bare metal. Kuma é implementado como um plano de controle sobre o Envoy e, como tal, pode instrumentar qualquer tráfego de layer 4/layer 7 para assegurar, observar, indicar e melhorar a conectividade entre serviços. A maioria das implementações de malha de serviços é direcionada nativamente no ecossistema Kubernetes, o que e si não é ruim, mas entrava a adoção de malha de serviço para aplicações não-Kubernetes existentes. Em vez de esperar por grandes esforços de transformação de plataforma serem completos, você pode agora usar Kuma e modernizar a infraestrutura de rede.

MicroK8s
AVALIE

Falamos sobre Kubernetes no passado e ele continua a ser a escolha padrão para implantação e gerenciamento de contêineres em clusters de produção. Contudo, está ficando cada vez mais difícil fornecer uma experiência offline similar para pessoas desenvolvedoras. Entre as opções, achamos MicroK8s bem útil. Para instalar MicroK8s snap, escolha um canal (stable, candidate, beta ou edge), e você pode rodar o Kubernetes com alguns poucos comandos. Você também pode acompanhar lançamentos e escolher atualizar sua configuração automaticamente.

Oculus Quest
AVALIE

Temos monitorado RA/RV (Realidade Aumentada/Virtual) em nosso Radar, mas seu apelo tem se limitado a plataformas específicas e opções de tethering. Oculus Quest muda o jogo, tornando-se um dos primeiros headsets de realidade virtual autônomos para o mercado de massa que não requer tethering ou suporte além de um smartphone. Esse dispositivo abre a porta para um grande avanço na potencial exposição de aplicações de realidade virtual, cuja demanda, por sua vez, levará o mercado a inovações mais agressivas. Aplaudimos a democratização da realidade virtual que esse dispositivo inaugura e mal podemos esperar para ver o que o futuro nos reserva.

ONNX
AVALIE

As ferramentas e ecossistemas de frameworks em torno de redes neurais estão se desenvolvendo rapidamente. A interoperabilidade entre elas, contudo, tem sido um desafio. Não é incomum na indústria de aprendizado de máquina prototipar e treinar rapidamente o modelo em uma ferramenta e então implementá-lo em uma ferramenta diferente para inferência. Como os formatos internos dessas ferramentas não são compatíveis, precisamos implementar e manter conversores confusos para deixar os modelos compatíveis. O formato Open Neural Exchange ONNX trata desse problema. No ONNX, as redes neurais são representadas como grafos, usando especificações de operador padrão e, junto com um formato de serialização para objetos treinados, modelos de redes neurais podem ser transferidos de uma ferramenta para outra. Isso abre muitas possibilidades, incluindo o Model Zoo, uma coleção de modelos pré-treinados no formato ONNX.

Contêineres sem raiz
AVALIE

Idealmente, contêineres devem ser gerenciados e executados pelo respectivo runtime do contêiner sem privilégios de raiz. Isso não é trivial, mas, quando alcançado, reduz a superfície de ataque e evita classes inteiras de

problemas de segurança, notadamente, escalonamento de privilégios fora do contêiner. A comunidade tem discutido isso como contêineres sem raiz há algum tempo, e é parte da especificação do tempo de execução de contêiner aberto e sua implementação padrão runc, que sustenta o Kubernetes. Agora, o Docker 19.03 apresenta contêineres sem raiz como uma funcionalidade experimental. Embora totalmente funcional, a funcionalidade não trabalha ainda com várias outras funcionalidades, como controles de recursos cgroups e perfis de segurança AppArmor.

Snowflake

AVALIE

Muitas vezes, relacionamos armazenamento de dados a uma infraestrutura central difícil de escalar e gerenciar com as crescentes demandas acerca dos dados. Snowflake, todavia, é uma nova solução de DataWarehousing SQL como serviço feita do zero para a nuvem. Com vários recursos cuidadosamente criados, como atomicidade em nível de banco, suporte de dados estruturado e semiestruturado, funções analíticas embutidas e, sobretudo, com uma clara separação de armazenamento, processamento e camadas de serviço, o Snowflake lida com a maioria dos desafios enfrentados em data warehousing.

Teleport

AVALIE

Teleport é um gateway de segurança para acessar remotamente infraestruturas em nuvem nativas. Uma de suas funcionalidades mais interessantes é sua habilidade de também atuar como um Certificate Authority (CA) para sua infraestrutura. Você pode emitir certificados de curta duração e construir um sofisticado controle de acesso baseado em papéis (RBAC) para sua infraestrutura Kubernetes (ou apenas para SSH). Com o aumento do foco na segurança da infraestrutura, é importante acompanhar as mudanças. Contudo, nem todos os eventos requerem o mesmo nível de auditoria. Com Teleport, você pode ficar com logging para a maioria dos eventos, mas ir um pouco além ao gravar a tela do usuário para sessões root mais privilegiadas.

PLATAFORMAS

Teleport é um gateway de segurança para acessar remotamente infraestruturas em nuvem nativas.

(Teleport)

FERRAMENTAS

Commitizen
ADOTE

Commitizen é uma ferramenta simples para ajudar a agilizar o processo de commit ao usar GIT. Ela induz o fornecimento de quaisquer campos requeridos e também formata sua mensagem de commit adequadamente. Diferentes convenções para descrever os formatos de check-in necessários são suportados e você pode adicionar o seu próprio por meio de um adaptador. Esta ferramenta simples economiza tempo e evita rejeições posteriores de um commit hook.

ESLint
ADOTE

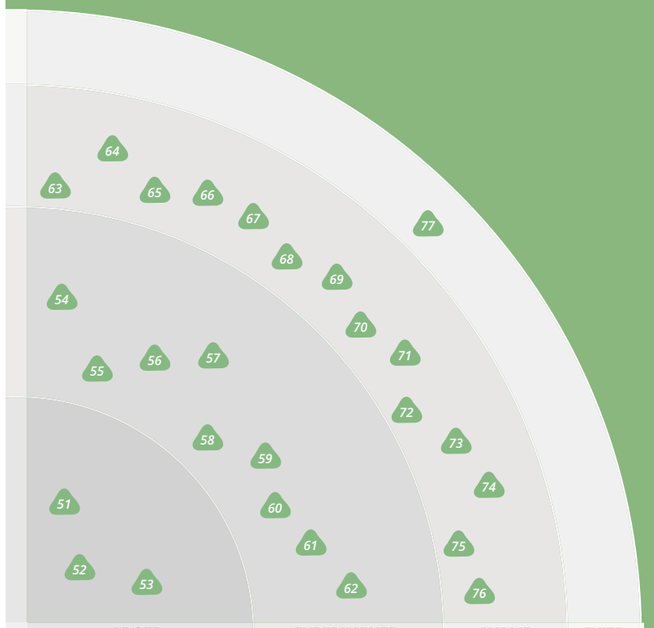
A ESLint está sendo usada como um padrão em muitos de nossos projetos. Como uma ferramenta linting para JavaScript, possui múltiplos conjuntos de regras, regras recomendadas e plugins para estender para frameworks ou estilos de JavaScript. Temos visto ela alavancar fortemente times na criação e execução de normas em seu código, ao permitir análise de código em tempo real durante o desenvolvimento. Pode ser usada para padronizar práticas de código, com a aplicação de boas práticas e estilização do código, e identificar vulnerabilidades em seu código. Ela faz isso ao se integrar bem com a maioria das IDEs e fornecer feedback em tempo real enquanto se está escrevendo o código. Suas regras de estilização, em particular, automaticamente corrigem os erros de linting, tornando o processo suave e eficaz sem incorrer em custo adicional de desenvolvimento. Pessoas desenvolvedoras podem rapidamente acelerar com as regras graças à documentação da comunidade, que faz um bom trabalho explicando padrões de código. À medida que a ESLint se torna mais comum e poderosa, tem ganhado força na indústria e isso é ilustrado pelo movimento do time do TypeScript para suportar e trabalhar com a ESLint em vez de investir em TSLint.

Guiador de Estilo React
ADOTE

React Styleguidist é um ambiente de desenvolvimento para componentes React. Ele inclui um servidor de desenvolvimento com capacidades de hot reloading e gera um guia de estilo HTML para compartilhar com os times. O guia de estilo mostra uma versão ao vivo de todos os componentes em um lugar com documentação de uso e uma lista de seus acessórios. Nós mencionamos o Guiador de Estilo React como um ambiente de desenvolvimento UI anteriormente, e com o tempo ele se tornou nossa escolha padrão entre ferramentas similares neste espaço.

Bitrise
EXPERIMENTE

Construir, testar e implantar aplicações móveis implica em passos complexos, especialmente quando consideramos uma pipeline que vai de repositório de código-fonte a lojas de aplicativos. Todos



ADOTE
51. Commitizen
52. ESLint
53. Guiador de Estilo React

EXPERIMENTE
54. Bitrise
55. Dependabot
56. Detekt
57. Figma
58. Jib
59. Loki
60. Trivy
61. Twistlock
62. Yocto Project

AVALIE
63. Aplas
64. asdf-vm
65. AWSume
66. dbt
67. Docker Notary
68. Facets
69. Falco
70. in-toto
71. Kubeflow
72. MemGuard
73. Open Policy Agent (OPA)
74. Pumba
75. Skaffold
76. What-If Tool

EVITE
77. Azure Data Factory para orquestração

FERRAMENTAS

Figma possui as mesmas funcionalidades de programas de design, como Sketch e Invision, mas permite colaboração com outra pessoa em tempo real.

(Figma)

Se você está construindo uma aplicação Java e usa Docker, você pode considerar usar Jib, do Google.

(Jib)

esses passos podem ser automatizados com scripts e pipelines de construção em ferramentas de CI/CD genéricas. Contudo, nossos times consideram Bitrise, uma ferramenta de entrega contínua específica de domínio para aplicações móveis, útil para aplicações móveis quando não há necessidade de integrar com pipelines de compilação para sistemas back-end. Bitrise é fácil de configurar e fornece um conjunto abrangente de passos pré-construídos para a maioria das necessidades de desenvolvimento móvel.

Dependabot

EXPERIMENTE

Manter dependências atualizadas é uma tarefa rotineira, mas por questões de segurança é importante responder a essas atualizações em tempo hábil. Você pode usar ferramentas para tornar esse processo o mais indolor e automatizado possível. Na prática, nossos times têm boas experiências com o Dependabot. Ele se integra com os repositórios do GitHub e verifica automaticamente dependências para novas versões. Quando requisitado, o Dependabot abrirá uma pull request com dependências atualizadas.

Detekt

EXPERIMENTE

Detekt é uma ferramenta de análise de código estático para Kotlin. Ela fornece uma análise do mau cheiro do código e relatórios de complexidade baseados em conjuntos de regras altamente configuráveis. Pode ser executada a partir da linha de comando e usando plugins, via Gradle, SonarQube e IntelliJ. Nossos times encontraram grande valor ao usar Detekt para manter a qualidade de código alta. Quando a análise e a geração de relatórios são integradas em uma pipeline de compilação, é obviamente importante que os relatórios sejam verificados regularmente e os times tirem um tempo para agir em seus achados.

Figma

EXPERIMENTE

Uma das grandes dores no design visual e de interação é a falta de ferramentas feitas para colaboração. É aí que a Figma entra. Ela tem as mesmas funcionalidades de programas de design, como Sketch e Invision, mas, ao proporcionar colaboração com outra pessoa ao mesmo tempo, ela ajuda na descoberta de novas ideias em conjunto, com capacidades de colaboração em tempo real. Nossos times acham a Figma muito útil, especialmente na capacitação e facilitação do trabalho de design remoto e distribuído. Além de suas capacidades de colaboração, a Figma também oferece uma API que ajuda a melhorar o processo de DesignOps.

Jib

EXPERIMENTE

Construir aplicações conteinerizadas pode demandar configurações complexas em ambientes de desenvolvimento e em agentes de build. Se você está construindo uma aplicação Java e usa Docker, você pode considerar usar Jib, do Google. Jib é um plugin de código aberto que suporta Maven e Gradle. O plugin Jib usa informações da sua configuração de compilação para construir sua aplicação diretamente como uma imagem Docker sem precisar de Dockerfile ou Docker daemon. O Jib otimiza a estratificação de imagens, prometendo acelerar as compilações subsequentes.

Loki

EXPERIMENTE

Loki é uma ferramenta de regressão visual que funciona com o Storybook, que mencionamos anteriormente no contexto de ambientes de desenvolvimento UI. Com poucas linhas de configuração, Loki pode ser usado para testar todos os componentes UI. O modo preferido de operação é usando o Chrome em um contêiner Docker, pois evita diferenças de um pixel quando os testes estão rodando em ambientes não-idênticos. Nossa experiência é de que os testes são bem estáveis, mas atualizações no Storybook tendem a fazer os teste falharem com pequenas diferenças. Também parece impossível testar componentes que usam `position: fixed`, mas você pode trabalhar nisso envolvendo o componente com um `fixed`.

Trivy

EXPERIMENTE

Pipelines de build que criam e implantam contêineres devem incluir escaneamento de segurança de contêiner. Nossos times gostam particularmente de Trivy, um scanner de vulnerabilidade para contêineres, porque é mais fácil de configurar do que outras ferramentas, graças ao seu envio como binário independente. Outros benefícios de Trivy são o fato de ser um software de código aberto e suportar contêineres distroless.

Twistlock

EXPERIMENTE

Twistlock é um produto comercial com detecção de vulnerabilidade de segurança durante o tempo de compilação e execução

e com recursos de prevenção. Esses recursos abrangem proteção de máquinas virtuais, agendadores de contêineres e contêineres para vários registros e repositórios com os quais as aplicações contam. Twistlock ajudou nossos times a acelerarem o desenvolvimento de aplicações regulamentadas, em que a infraestrutura e a arquitetura da aplicação requerem conformidade com, por exemplo, padrões de Payment Card Industry (PCI) e da Health Insurance Portability and Accountability Act (HIPAA). Nossos times aproveitaram a experiência de desenvolvimento que o Twistlock fornece: a capacidade de executar provisionamentos como código, a fácil integração com outras plataformas comuns de observabilidade e benchmarks incluídos para mensurar a infraestrutura em relação às boas práticas que são consenso na indústria. Executamos Twistlock com escaneamentos regulares de tempo de execução em nossas aplicações nativas em nuvem, especificamente quando a conformidade regulatória é requerida.

Projeto Yocto

EXPERIMENTE

Cada vez mais estamos vendo dispositivos poderosos de Internet das Coisas que rodam Linux em vez de um sistema operacional especial embutido. Para reduzir o uso de recursos e diminuir a superfície de ataque, faz sentido construir uma distribuição Linux personalizada que contenha apenas as ferramentas e dependências necessárias para rodar o software naquele dispositivo. Neste contexto, o Projeto Yocto renovou sua relevância como ferramenta para criar uma distribuição Linux feita sob medida para as necessidades de casos específicos. A curva de aprendizado é acentuada e, devido à sua flexibilidade, pode ser fácil fazer a coisa errada. Contudo, ao longo de seus muitos anos de existência, o projeto Yocto atraiu uma comunidade ativa que pode ajudar. Comparando a ferramentas similares, é mais fácil integrar a um fluxo de trabalho de entrega contínua e, diferente do Android Things ou Ubuntu core, por exemplo, não está atrelado a um ecossistema específico.

Aplas

AVALIE

Frequentemente é muito difícil controlar as propriedades de nossos softwares à medida que elas ficam mais complexas. Aplas é uma nova ferramenta de mapeamento de software que pode ser usada para criar visualizações de cenários de nosso software na forma de mapas. A ferramenta trabalha ingerindo metadados sobre seus sistemas existentes e então mostrando um mapa sobre o qual várias visualizações podem ser projetadas. A ingestão pode tanto ser um processo manual como automatizado por meio de APIs. Estamos muito otimistas vendo esse produto evoluir e vislumbrando o que é possível com a coleta automatizada de metadados. Deveria ser possível, por exemplo, expor funções de aptidão arquitetônica, como custo de execução para criar visualizações do quanto está sendo gasto em infraestrutura de nuvem. Entender quais sistemas conversam com outros sistemas por meio de qual tecnologia é outro problema que frequentemente enfrentamos, e o Aplas pode visualizar isso para nós.

asdf-vm

AVALIE

asdf-vm é uma ferramenta de linha de comando para gerenciar versões de execução de múltiplas linguagens por projeto. É similar a outras ferramentas de gerenciamento por linha de comando, como RVM para Ruby e nvm para Node.js, com a vantagem de um arquitetura de plugin extensível para lidar com múltiplas linguagens. Sua lista atual de plugins inclui muitas linguagens e também ferramentas, como Bazel ou tflint, cuja versão de execução você talvez precise gerenciar por projeto.

AWSume

AVALIE

AWSume é um script conveniente para gerenciar tokens de sessão AWS e aceitar credenciais a partir da linha de comando. Achamos o AWSume muito útil quando lidamos com múltiplas contas AWS ao mesmo tempo. Em vez especificar roles individualmente em cada comando, o script lê a partir do cache do CLI e exporta para variáveis de ambiente. Como resultado, tanto os comandos quanto os SDKs AWS encontram as credenciais corretas.

dbt

AVALIE

A transformação de dados é uma parte essencial dos fluxos de trabalho de processamento de dados: filtra, agrupa ou reune múltiplas fontes em um formato adequado para analisar dados ou alimentar modelos de aprendizado de dados. dbt é uma ferramenta de código aberto e um produto SaaS comercial que fornece capacidades de transformação simples e efetivas para analistas de dados. Os frameworks atuais e o ferramental para transformação de dados caem ou no grupo de *poderosos e flexíveis* – requisitando um entendimento íntimo do modelo de programação e linguagens do framework, tais como Apache Spark – ou no grupo

FERRAMENTAS

O Projeto Yocto renovou sua relevância como ferramenta para criar uma distribuição Linux feita sob medida para as necessidades de casos específicos, como por exemplo Internet das Coisas.

(Projeto Yocto)

FERRAMENTAS

Aplas é uma nova ferramenta de mapeamento de software que pode ser usada para criar visualizações de cenários de nosso software na forma de mapas.

(Aplas)

das ferramentas bobas de UI, de arrastar e soltar, que não se prestam a práticas de engenharia confiáveis, tais como testes automatizados e implantação. dbt preenche um nicho: usa SQL – uma interface amplamente entendida – para modelar simples transformações em lote, enquanto fornece ferramentas de linha de comando que incentivam boas práticas de engenharia, como versionamento, testes automatizados e implantações. Essencialmente, implementa a modelagem de transformação baseada em SQL como código. dbt atualmente suporta múltiplas fontes de dados, incluindo Snowflake e Postgres, e fornece várias opções de execução, como a Airflow e a própria oferta de nuvem da Apache. Sua capacidade de transformação está limitada ao que o SQL oferece e, até a publicação deste texto, não suporta transformações de streaming em tempo real.

Docker Notary

AVALIE

Docker Notary é uma ferramenta de código aberto que permite assinar recursos como imagens, arquivos e contêineres. Isso significa que a procedência dos recursos pode ser declarada, o que é superútil em ambientes regulados e para boas práticas em todos os lugares. Como exemplo, quando um contêiner é criado, ele é assinado por uma chave privada e um hash, ligados à identidade da pessoa que publicou e armazenados como metadata. Uma vez publicado, a procedência do contêiner (ou outro recurso) pode ser verificada usando-se o hash da imagem e a chave pública de quem publicou. Há registros confiáveis disponíveis publicamente, como o Docker Trusted Registry, mas é possível rodar o seu próprio. Nossos times notaram algumas dificuldades rodando servidores Notary locais e sugerem usar um registro que inclua o Notary onde for possível.

Facets

AVALIE

Dada a crescente quantidade de decisões ponderadas derivadas de grandes conjuntos de dados, seja diretamente ou como um input de treinamento para modelos de aprendizado, é importante entender as lacunas, falhas e potenciais vieses em seus dados. O projeto Facets do Google fornece duas ferramentas úteis neste espaço: Facets Overview e Facets Dive. A Facets Overview visualiza a distribuição de valores para features em um conjunto de dados, pode mostrar vieses nos conjuntos de treinamento e de validação e pode ser usada para comparar múltiplos conjuntos de dados. A Facets Dive é para detalhar e visualizar pontos de dados individuais em grandes conjuntos de dados, usando diferentes dimensões visuais para explorar as relações entre os atributos. As duas são úteis para a execução de testes de viés ético.

Falco

AVALIE

Com a adoção crescente de Kubernetes como um orquestrador de contêineres, o conjunto de ferramentas de segurança acerca de contêineres e Kubernetes está evoluindo rapidamente. Falco é uma das ferramentas nativas de contêiner que visa lidar com segurança do tempo de execução. Falco alavanca a instrumentação de kernel do Linux do Sysdig e profiling de chamadas de sistema, e nos permite obter grandes insights a respeito do comportamento do sistema, ajudando a detectar atividades anormais em aplicações, contêineres, hosts subjacentes ou no orquestrador Kubernetes em si. Gostamos da capacidade de Falco de detectar ameaças sem injetar códigos de terceiros ou contêineres sidecar.

in-toto

AVALIE

Estamos vendo o uso crescente do atestado binário para deixar segura a cadeia de suprimentos de software, especificamente em indústrias reguladas. As abordagens favorecidas atualmente parecem envolver tanto a construção de um sistema personalizado para implementar a verificação binária, quanto a dependência de um fornecedor de serviço de nuvem. Estamos otimistas de ver o código aberto in-toto entrar neste espaço. O in-toto é um framework para verificar criptograficamente cada componente e passo do trajeto de produção de um artefato de software. O projeto inclui um número de integrações em muitas ferramentas de compilação, auditoria de contêineres e implantação usadas amplamente. Uma ferramenta de cadeia de suprimentos de software pode ser uma parte crítica do aparato de segurança de uma empresa, então gostamos de que, como um projeto de código aberto, o comportamento do in-toto seja transparente, e sua própria integridade e cadeia de suprimentos podem ser verificadas pela comunidade. Teremos que esperar para ver se a ferramenta ganhará uma massa crítica de usuários e contribuidores para competir neste espaço.

Kubeflow
AVALIE

Kubeflow é interessante por dois motivos. Primeiro, é um uso inovador de Kubernetes Operators, que destacamos em nossa edição de abril de 2019 do Radar. Segundo, fornece uma maneira de codificar e versionar fluxos de trabalho de aprendizado de máquina para que eles sejam mais facilmente transportados de um ambiente de execução para outro. O Kubeflow consiste de vários componentes, incluindo notebooks Jupyter, pipelines de dados e ferramentas de controle. Muitos desses componentes são empacotados como operadores Kubernetes para aproveitar a capacidade do Kubernetes de reagir a eventos gerados por pods implementando vários estágios do fluxo de trabalho. Ao empacotar os programas individuais e os dados como contêineres, fluxos de trabalho inteiros podem ser transportados de um ambiente para outro. Isso pode ser útil quando mover um fluxo de trabalho útil, porém computacionalmente desafiador e desenvolvido na nuvem, para um supercomputador personalizado ou cluster de uma unidade de processamento tensorial.

MemGuard
AVALIE

Se sua aplicação lida com informação sensível (como chaves criptográficas) como texto simples em memória, há uma alta probabilidade de que alguém possa explorá-la como um vetor de ataque e comprometer a informação. A maioria das soluções baseadas em nuvem frequentemente usa módulos de segurança de hardware (HSM) para evitar tais ataques. Contudo, se você está em uma situação em que precisa fazer isso de maneira auto-hospedada sem acesso a HSM, então achamos que o MemGuard pode ser bem útil. O MemGuard age como um enclave de software de segurança para armazenar informação sensível em memória. Apesar de MemGuard não ser um substituto para os HSMs, ele traz uma quantia de táticas de segurança, tais como proteção contra ataques cold boot, evitando interferência na coleta do lixo e fortalecendo guard pages para reduzir a probabilidade de dados sensíveis serem expostos.

Open Policy Agent (OPA)
AVALIE

Definir e executar políticas de segurança uniformemente em um cenário de tecnologia diverso é um desafio. Mesmo para aplicações simples, você tem que controlar o acesso a seus componentes – tais como orquestradores de contêineres, serviços e armazenamento de dados para manter o estado do serviço – usando sua configuração de política de segurança embutida dos componentes e mecanismos de execução.

Estamos otimistas com o Open Policy Agent (OPA), uma tecnologia de código aberto que tenta resolver esse problema. O OPA deixa você definir controle de acesso refinado e políticas flexíveis como código, usando a linguagem de definição de política Rego. Rego executa as políticas de maneira distribuída e limpa fora do código da aplicação. À época em que escrevemos, o OPA implementou a definição de política uniforme e flexível e execução para assegurar o acesso a APIs do Kubernetes, APIs de microsserviços por meio do sidecar do Envoy e Kafka. Também pode ser usado como sidecar para qualquer serviço para verificar acesso a políticas ou filtrar dados de resposta. Styra, a empresa por trás do OPA, fornece soluções comerciais para visibilidade centralizada para políticas distribuídas. Gostamos de ver o OPA amadurecer por meio do programa de incubação CNCF e continuar fornecendo suporte para cenários de execução de políticas mais desafiadores, tais como armazéns de dados diversos.

Pumba
AVALIE

Pumba é uma ferramenta de teste de caos e emulação de rede para Docker. O Pumba pode eliminar, parar, remover ou pausar contêineres Docker. Pode também emular redes e simular diferentes falhas de rede, como atrasos, perda de pacotes e limites na largura de banda. O Pumba usa a ferramenta tc para emulação de rede, o que significa que precisa estar disponível em nossos contêineres ou precisamos rodar o Pumba em um contêiner sidecar com tc. Pumba é particularmente útil quando queremos executar testes de caos automatizados em um sistema distribuído rodando em vários contêineres localmente ou na pipeline embutida.

Skaffold
AVALIE

O Google nos traz a Skaffold, uma ferramenta de código aberto para automatizar fluxos de trabalho de

FERRAMENTAS

Com a adoção crescente de Kubernetes como um orquestrador de contêineres, o conjunto de ferramentas de segurança acerca de contêineres e Kubernetes está evoluindo rapidamente. Falco é uma das ferramentas nativas de contêiner que visa lidar com segurança do tempo de execução.

(Falco)

FERRAMENTAS

What-If Tool ajuda cientistas de dados a explorar o comportamento de um modelo e visualizar o impacto que várias funcionalidades e conjuntos de dados têm no resultado.

(What-If Tool)

desenvolvimento locais, incluindo deploy em Kubernetes. A Skaffold detecta mudanças no código-fonte e ativa fluxos de trabalho para compilar, taggear e implantar em um cluster K8s, incluindo a captura de logs de aplicações de volta para a linha de comando. Os fluxos de trabalho são plugáveis com diferentes ferramentas de compilação e implantação, mas isso acontece com uma configuração padrão opinativa para tornar mais fácil a inicialização.

What-If Tool

AVALIE

O mundo do aprendizado de máquina mudou levemente sua ênfase de explorar do que os modelos são capazes de entender para como eles fazem isso. Preocupações sobre introdução de vieses ou generalizar demais a aplicabilidade do modelo resultou em novas ferramentas interessantes, tais como What-if Tool (WIT). Essa ferramenta ajuda cientistas de dados a mergulhar no comportamento de um modelo e visualizar o impacto que várias funcionalidades e conjuntos de dados têm no resultado. Introduzida pelo Google e disponível tanto por meio do Tensorboard ou de notebooks Jupyter, WIT simplifica as tarefas de comparar modelos, partir conjuntos de dados, visualizar facetas e editar pontos de dados individuais. Apesar de WIT simplificar a execução dessas análises, ainda é necessário um profundo entendimento de matemática e teoria por trás dos modelos. É uma ferramenta para cientistas de dados obterem insights mais profundos sobre o comportamento do modelo. Usuários ingênuos não devem esperar que qualquer ferramenta remova o risco ou minimize os danos resultantes de um algoritmo mal treinado.

Azure Data Factory para orquestração

EVITE

Azure Data Factory (ADF) é atualmente o produto padrão da Azure para orquestrar pipelines de processamento de dados. Ele suporta ingestão de dados, copiar dados de e para diferentes tipos de armazenamento localmente ou no Azure e executar lógica de transformação. Embora tenhamos tido uma experiência razoável com ADF para migrações simples de armazéns de dados de local para nuvem, desencorajamos o uso de Azure Data Factory para orquestração para pipelines complexas de processamento de dados. Nossa experiência tem sido desafiadora por vários fatores, incluindo cobertura limitada de capacidades que podem ser implementadas por meio de código primeiro, pois parece que ADF está priorizando a possibilidade das capacidades de plataforma de baixo código; capacidade ruim de debug e relatar erros; observação limitada, já que as capacidades de login da ADF não se integram com outros produtos, como Azure Data Lake Storage ou Databricks, tornando difícil obter uma observação de ponta-a-ponta; e disponibilidade de mecanismos de acionamento da fonte de dados apenas para certas regiões. Nesse momento, encorajamos usar outras ferramentas de orquestração de código aberto (ex.: Airflow) para pipelines de dados complexos, e limitar ADF para cópias de dados ou snapshotting. Esperamos que a ADF aborde essas preocupações para suportar fluxos de trabalho mais complexos de processamento de dados e priorize o acesso a capacidade por meio de código primeiro.

LINGUAGENS & FRAMEWORKS

Arrow
EXPERIMENTE

Arrow é uma biblioteca de programação funcional para Kotlin, criada a partir da fusão de duas bibliotecas populares já existentes (kategory e funKTionale). Enquanto Kotlin fornece peças para a programação funcional, Arrow entrega para pessoas desenvolvedoras de aplicações um pacote de abstrações de alto nível pronto para uso. Ele fornece tipos de dados, tyope classes, effects, optics e outros padrões de programação funcional, assim como integração com bibliotecas populares. Nossas impressões positivas iniciais do Arrow foram confirmadas quando o usamos para construir aplicações que estão agora em produção.

Flutter
EXPERIMENTE

Muitos de nossos times usam Flutter e realmente gostam. É um framework multiplataforma que permite que você escreva aplicativos mobile nativos em Dart. Ele se beneficia do Dart, pode ser compilado para código nativo e se comunica com a plataforma-alvo sem bridge e mudança de contexto. A funcionalidade de hot reload é impressionante e fornece feedback visual super-rápido quando editamos código. Estamos confiantes em recomendar que você experimente o Flutter em algum de seus projetos.

jest-when
EXPERIMENTE

jest-when é uma biblioteca leve em JavaScript que complementa a Jest combinando argumentos na chamada de funções mock. Jest é uma ótima ferramenta para testar a stack, e jest-when permite que você espere argumentos específicos para funções mock. Dessa forma, permite que você escreva teste unitários de módulos com muitas dependências mais robustos.

Micronaut
EXPERIMENTE

Micronaut é um framework JVM para criação de serviços usando Java, Kotlin ou Groovy. Ele se distingue por seu consumo pequeno de memória e sua rápida inicialização. Ele consegue essas melhorias ao evitar o uso de reflexão em tempo de execução para injeção de dependência (DI) e geração de proxy, o que é comum em frameworks tradicionais. Em vez disso, usa um contêiner DI/AOP, que executa a injeção de dependência no momento da compilação. Isso o torna atrativo não apenas para microsserviços tradicionais implantados em servidores, como também no contexto de, por exemplo, Internet das Coisas (IoT), aplicativos Android e funções sem servidor. O Micronaut é uma opção de entrada muito promissora para o espaço dos frameworks full-stack para a plataforma JVM, e o estamos vendo em cada vez mais projetos em produção, o que nos leva a movê-lo para o anel Experimente.

ADOTE

EXPERIMENTE
78. Arrow
79. Flutter
80. jest-when
81. Micronaut
82. React Hooks
83. Biblioteca de Testes para React
84. Componentes estilizados
85. Tensorflow

AVALIE
86. Fairseq
87. Flair
88. Gatsby.js
89. GraphQL
90. KotlinTest
91. NestJS
92. Paged.js
93. Quarkus
94. SwiftUI
95. Testcontainers

EVITE
96. Enzyme

LANGUAGES & FRAMEWORKS

Micronaut é um framework JVM para criação de serviços usando Java, Kotlin ou Groovy. Ele se distingue por seu consumo pequeno de memória e sua rápida inicialização.

(Micronaut)

A Biblioteca de Testes para React superou as alternativas se tornando o padrão mais aceito para testes de front-end baseados em React.

(Biblioteca de Testes para React)

React Hooks
EXPERIMENTE

No começo deste ano, o React Hooks foi introduzido ao popular framework JavaScript. Eles tornaram possível usar estado e outras funcionalidades do React sem escrever uma classe, oferecendo uma abordagem mais limpa do que componentes de ordem superior ou render-props para casos de uso. Bibliotecas, tais como Material UI e Apollo já mudaram para o Hooks. Há alguns problemas com testes Hooks, principalmente com Enzyme, que contribuíram para nossa reavaliação da Enzyme como escolha de ferramenta.

Biblioteca de Testes para React
EXPERIMENTE

O mundo do JavaScript se move muito rápido e, à medida que ganhamos mais experiência usando um framework, nossas recomendações mudam. A Biblioteca de Testes para React é um bom exemplo de um framework que, com profundo uso, ocultou as alternativas para se tornar o padrão mais razoável para testes de frontend baseados em React. Nossos times gostam do fato de que teste escritos com esse framework são menos frágeis do que com frameworks alternativos, como o Enzyme, porque ele te encoraja a testar relações de componentes individualmente, em vez de testar todos os detalhes de implementação.

Componentes estilizados
EXPERIMENTE

Usar componentes estilizados torna possível adicionar o CSS necessário para estilizar um componente React diretamente no código JavaScript que cria o componente. Isso reduz enormemente o sofrimento com gerenciamento de CSS e evita a necessidade de convenções de nomenclatura ou outro meio de evitar conflitos de nomenclatura em CSS. Pessoas desenvolvedoras podem ver a estilização ao olharem para a definição do componente e não é necessário memorizar vários megabytes de CSS. Claro, colocar o CSS dentro do código JavaScript pode dificultar a obtenção de uma visão consistente da estilização de diferentes componentes, por isso recomendamos entender as vantagens e desvantagens dessa abordagem.

Tensorflow
EXPERIMENTE

Com seu lançamento 2.0, o TensorFlow mantém sua proeminência como o framework de aprendizado de máquina líder da indústria. O TensorFlow começou como um pacote de processamento numérico que gradualmente se expandiu para incluir bibliotecas que suportavam uma variedade de abordagens de aprendizado de máquina e ambientes de execução, variando de CPU mobile para grandes clusters de GPU. Ao longo do caminho, uma grande quantidade de frameworks ficou disponível para simplificar as tarefas de criação de redes e treinamento. Ao mesmo tempo, outros frameworks, notadamente o PyTorch, ofereciam um modelo de programação crucial que deixavam o debugging e execução mais simples e mais fáceis. O TensorFlow 2.0 agora tem como padrão o fluxo imperativo (execução eager) e adota a Keras como única API de alto nível. Embora essas mudanças modernizem a usabilidade do TensorFlow e o deixem mais competitivo com o PyTorch, é uma reescrita significativa que muitas vezes tem problemas de compatibilidade – muitas ferramentas e frameworks auxiliares no ecossistema do TensorFlow não funcionarão imediatamente com a nova versão. Por enquanto, considere se você quer projetar e experimentar no TensorFlow 2.0, mas reverta para a versão 1 para executar seus modelos em produção.

Fairseq
AVALIE

Fairseq é um conjunto de ferramentas de modelagem Seq2Seq do time de Pesquisa de IA do Facebook, que permite que pesquisadores e desenvolvedores treinem modelos personalizados para tradução, sumarização, modelagem de linguagem e outras tarefas de NLP (processamento de linguagem natural). Para usuários de PyTorch, é uma boa escolha. Ele fornece implementações de referência de vários modelos Seq2Seq, suporta treinamento distribuído por múltiplos GPUs e máquinas, é bastante extensível e tem vários modelos pré-treinados, incluindo RoBERTa, que é uma otimização em cima do BERT.

Flair
AVALIE

Flair é um framework simples baseado em Python para processamento de NLP. Permite aos usuários executarem tarefas de NLP padrão, tais como reconhecimento de entidade mencionada (NER), marcação de parte do discurso (PoS), classificação e desambiguação do sentido da palavra, e tem um bom desempenho com uma gama de tarefas NLP. O Flair apresenta uma interface simples e unificada para uma variedade de palavras e documentos incorporados, incluindo BERT, Elmo e suas próprias incorporações Flair. Também tem suporte multi-idioma. O framework em si é

construído em cima do PyTorch. Estamos usando em alguns de nossos projetos e gostamos da sua facilidade de uso e abstrações poderosas.

Gatsby.js
AVALIE

Gatsby.js é um framework para escrever aplicações para web em um estilo de arquitetura conhecido como JAMstack. Parte da aplicação é gerada em tempo de build e implantada como um site estático, enquanto o restante da funcionalidade é implementado como uma aplicação web progressiva (PWP) rodando no navegador. Tais aplicações trabalham sem o código rodando no lado do servidor. Normalmente, no entanto, a aplicação web progressiva faz chamadas para APIs e soluções SaaS de terceiros para gerenciamento de conteúdo, por exemplo. No caso do Gatsby.js, todo o código do cliente e do tempo de compilação é escrito usando React. O framework inclui algumas otimizações pra deixar a aplicação web mais rápida. Fornece código e divisão de dados out-of-the-box para minimizar tempo de carregamento e acelera a performance quando navegando na aplicação utilizando pré-carregamento. APIs são requisitadas via GraphQL e vários plugins simplificam a integração com serviços existentes.

GraphQL
AVALIE

Temos visto muitas implementações bem-sucedidas com GraphQL em nossos projetos. Temos vistos também alguns padrões de uso interessantes, incluindo GraphQL para agregação de recursos do lado do servidor. Dito isso, nos preocupamos com o uso equivocado desse framework e alguns dos problemas que podem ocorrer. Exemplos incluem problemas de performance com consultas N+1 e um excesso de código necessário quando adicionamos novos modelos, levando à complexidade. Há maneiras de se contornar esses problemas, como por exemplo o uso de cache de consulta. Mesmo que GraphQL não seja uma solução mágica, ainda achamos que é válido avaliar como parte de sua arquitetura.

KotlinTest
AVALIE

KotlinTest é uma ferramenta autônoma de testes para o ecossistema Kotlin, da qual nossos times gostaram. Ela permite testes baseados em propriedade, uma técnica que destacamos no Radar anterior. As principais vantagens são sua oferta variada de estilos de teste para estruturar as suítes de teste e seu conjunto abrangente de *matchers*, o que permite testes expressivos em uma elegante DSL interna.

NestJS
AVALIE

NestJS é um framework Node.js do lado do servidor escrito em TypeScript. Ao integrar a rica ecologia da comunidade Node.js, o NestJS fornece uma arquitetura de aplicação out-of-the-box. O modelo mental para desenvolver em NestJS é similar à versão do lado do servidor do Angular ou da versão TypeScript do Spring Boot, então a curva de aprendizado para pessoas desenvolvedoras é baixa. O NestJS suporta protocolos como GraphQL, Websocket e bibliotecas ORM.

Paged.js
AVALIE

Ao usar HTML e tecnologias relacionadas para produzir livros e outros impressos, a questão da paginação deve ser considerada. Isso inclui contadores de páginas, elementos repetidos no cabeçalho e rodapé, assim como mecanismos para evitar quebras de página estranhas. Paged.js é uma biblioteca de código aberto que implementa uma série de polyfills para módulos de CSS para mídia paginada e conteúdo gerado para mídia paginada. É ainda experimental, mas preenche um espaço importante na história "escreva uma vez, publique em todos os lugares" para HTML.

Quarkus
AVALIE

Quarkus é um framework com foco em contêineres, nativo de nuvem, criado pela Red Hat para aplicações escritas em Java. Tem um tempo de inicialização muito rápido (dezenas de milissegundos) e utiliza pouca memória, o que o torna um bom candidato para FaaS ou frequente escalonamento para cima ou para baixo em um orquestrador de contêiner. Assim como o Micronaut, o Quarkus consegue isso usando técnicas de compilação à frente do tempo para fazer injeção de dependência no momento da compilação e evitar os custos de execução de reflexão. Ele também funciona bem com o Native Image do GraalVM, que reduz ainda mais o tempo de inicialização. O Quarkus suporta tanto

LANGUAGES & FRAMEWORKS

Gatsby.js é um framework para escrever aplicações para web em um estilo de arquitetura conhecido como JAMstack. Ele fornece código e divisão de dados out-of-the-box para minimizar tempo de carregamento e acelera a performance quando ao navegar na aplicação utilizando recursos de pré-carregamento.

(Gatsby.js)

LANGUAGES & FRAMEWORKS

Quarkus é um framework com foco em contêineres, nativo de nuvem, criado pela Red Hat para aplicações escritas em Java. Tem um tempo de inicialização muito rápido e utiliza pouca memória.

(Quarkus)

modelos imperativos como reativos. Junto com o Micronaut e o Helidon, Quarkus está liderando a nova geração de frameworks Java, que tentam resolver a performance de inicialização e memória sem sacrificar a eficácia de desenvolvimento. Ganhou muita atenção da comunidade e vale a pena observar de perto.

SwiftUI
AVALIE

A Apple deu um passo enorme em direção ao seu novo framework SwiftUI para implementar interfaces de usuário em plataformas macOS e iOS. Gostamos do fato de que SwiftUI vai além da relação um tanto quanto deselegante entre Interface Builder e XCode e adota uma abordagem coerente, declarativa e centrada no código. Você agora pode ver seu código e a interface visual resultante lado a lado no XCode 11, tornando a experiência de desenvolvimento muito melhor. O framework SwiftUI também traz inspiração do mundo do React.js, que tem dominado o desenvolvimento web nos últimos anos. Valores imutáveis em modelos de visualização e um mecanismo de atualização assíncrono criam um modelo de programação reativo unificado. Isso dá às pessoas desenvolvedoras uma alternativa totalmente nativa para frameworks reativos similares, tais como React Native ou Flutter. Apesar de SwiftUI definitivamente representar o futuro do desenvolvimento UI da Apple, é bem novo e levará tempo para lapidar as arestas. Esperamos por uma documentação melhorada e uma comunidade de pessoas desenvolvedoras que possa estabelecer um conjunto de práticas para testes e abordar outras preocupações de engenharia.

Testcontainers
AVALIE

Criar ambientes confiáveis para executar testes automatizados é um problema perene, particularmente porque o número de componentes de que os sistemas modernos dependem continuam aumentando. Testcontainers é uma biblioteca Java que ajuda a mitigar esse desafio ao gerenciar dependências 'dockerizadas' para seus testes. Isso é particularmente útil para gerar instâncias de base de dados repetíveis ou infraestrutura similar, mas também pode ser usada em navegadores web para testes UI. Nossos times consideram essa biblioteca útil para deixar testes de integração mais confiáveis com esses contêineres programáveis, leves e descartáveis.

Enzyme
EVITE

Nós nem sempre movemos ferramentas descontinuadas para Evite no Radar, mas nossos times sentem que a Enzyme foi substituída na construção de testes de unidade de componentes UI React pela Biblioteca de Teste React. Times usando Enzyme acham que seu foco nos testes de componentes internos leva a testes frágeis e insustentáveis.

Quer se atualizar com artigos e informações relacionadas ao radar?
Siga a gente nas redes sociais ou torne-se assinante.

assine

ThoughtWorks®

Somos uma consultoria global de software e uma comunidade de pessoas apaixonadas e guiadas por propósitos. Pensamos de maneira disruptiva para entregar tecnologia capaz de enfrentar os desafios mais difíceis de nossas clientes, ao mesmo tempo em que buscamos revolucionar a indústria de TI e provocar uma mudança social positiva.

Fundada há 25 anos, a ThoughtWorks se tornou uma empresa com mais de 7000 pessoas, incluindo uma área de produtos que desenvolve ferramentas pioneiras para times de software. A ThoughtWorks tem 43 escritórios em 14 países: Alemanha, Austrália, Brasil, Canadá, Chile, China, Equador, Espanha, Estados Unidos, Índia, Itália, Reino Unido, Singapura e Tailândia.

thoughtworks.com

ThoughtWorks®

thoughtworks.com/radar

#TWTechRadar