ThoughtWorks®

TECHNOLOGY RADAR

Um guia com opiniões firmes
sobre as fronteiras da tecnologia

Vol.22

thoughtworks.com/radar
#TWTechRadar

Contribuições

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

O Conselho Consultivo de Tecnologia (TAB) é um grupo de aproximadamente 20 tecnologistas seniores da ThoughtWorks. O TAB se reúne duas vezes por ano pessoalmente e quinzenalmente por videochamada. Seu papel principal é ser um grupo consultivo para a CTO da ThoughtWorks, Rebecca Parsons.

O TAB atua examinando tópicos que afetam tecnologias e tecnologistas da ThoughtWorks. Normalmente, criamos o Radar em reuniões presenciais, mas, dada a pandemia global que vivemos, este é o primeiro Technology Radar a ser criado por meio de um evento virtual.

Rebecca Parsons (CTO)

Martin Fowler (Chief Scientist)

Bharani Subramaniam

Birgitta Böckeler

Camilla Crispim

Erik Dörnenburg

Evan Bottcher

Fausto de la Torre

Hao Xu

Ian Cartwright

James Lewis

Jonny LeRoy

Lakshminarasimhan Sudarshan

Mike Mason

Neal Ford

Ni Wang

Rachel Laycock

Scott Shaw

Shangqi Liu

Zhamak Dehghani

© ThoughtWorks, Inc. Todos os direitos reservados.

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.

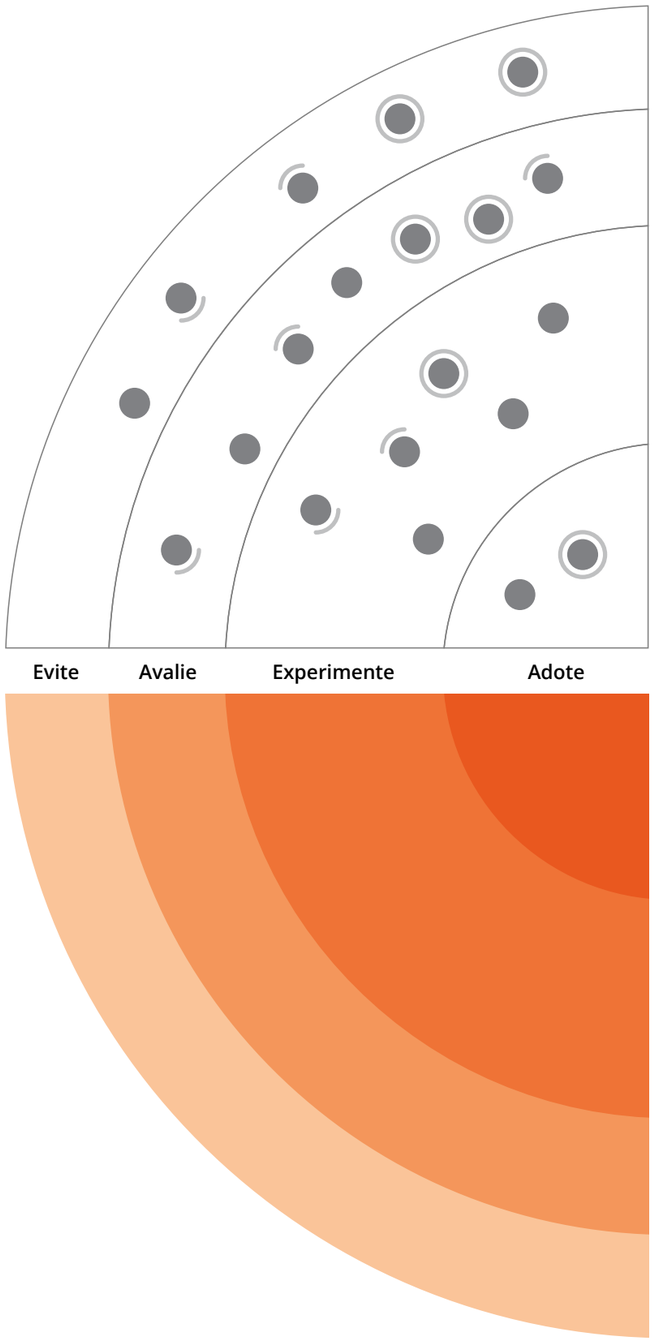
Para mais informações sobre o Radar, veja: thoughtworks.com/radar/faq

© ThoughtWorks, Inc. Todos os direitos reservados.

Radar em um relance

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

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



Novo

Modificado

Sem modificação

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

Adote

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

Experimente

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

Avalie

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

Evite

Prossiga com cautela.

© ThoughtWorks, Inc. Todos os direitos reservados.

Temas desta edição

O elefante na sala do Zoom

"A necessidade é a mãe da invenção"
— Provérbio

Muitas empresas vinham testando a ideia do trabalho remoto à medida que a tecnologia que o habilita amadurecia. Mas, de repente, uma pandemia global obrigou empresas de todo o mundo a mudar rápida e fundamentalmente sua maneira de trabalhar para de alguma forma preservar a produtividade. Como muitas pessoas já observaram, "trabalhar em casa" é totalmente diferente de "obrigatoriamente trabalhar em casa durante uma pandemia", e nós achamos que há pela frente uma longa jornada em busca da produtividade integral nesse novo contexto. Nunca acreditamos que a criação remota de uma edição do Radar fosse possível e, no entanto, aqui estamos — este é o primeiro Radar que produzimos sem nos encontrar pessoalmente. Muitos dos blips propostos abordam a necessidade urgente de permitir a colaboração remota de primeira classe. Não queríamos ignorar o elefante na sala e não comentar a crise, mas realizar um bom trabalho de colaboração remota é um assunto complexo e cheio de nuances, e certamente nem todos os nossos conselhos sobre o tema se encaixariam no formato do Radar. Portanto, junto a esta edição, você encontrará um podcast em que discutimos nossas experiências na criação remota do Radar, um relatório de experiências incluindo conselhos sobre produtividade trabalhando remotamente, um webinar sobre estratégias tecnológicas durante a crise e links para outros materiais da ThoughtWorks, incluindo nosso guia para trabalho remoto. Esperamos que estes, assim como outros materiais disponíveis na Internet, possam ajudar as organizações que tentam navegar por essas águas desconhecidas.

X também é software

Frequentemente, incentivamos outras partes do ecossistema de entrega de software a adotar práticas de engenharia benéficas, pioneiramente lideradas por times de desenvolvimento ágil de software. Voltamos a esse tópico com muita frequência porque continuamos a encontrar nichos em que vemos um progresso lento nessa área. Para este Radar, decidimos destacar novamente a infraestrutura como código, bem como pipelines como código, e também tivemos várias conversas sobre configurações de infraestrutura, pipelines de aprendizado de máquina e outras áreas relacionadas. Concluímos que os times que geralmente são responsáveis por essas áreas não adotam práticas perenes de engenharia, como a aplicação dos princípios de design de software, automação, integração contínua, testes e assim por diante. Entendemos que muitos fatores dificultam o movimento rápido em algumas práticas de engenharia: complexidade (tanto a essencial quanto a acidental), falta de conhecimento, impedimentos políticos, falta de ferramentas adequadas e muitos outros. No entanto, os benefícios para as organizações que adotam práticas ágeis de entrega de software são nítidos e valem o esforço para serem alcançados.

Perspectivas de dados amadurecendo e expandindo

Um tema que abrangeu muitos blips e quadrantes nesta edição diz respeito à maturidade dos dados, particularmente, técnicas e ferramentas que envolvem dados analíticos e aprendizado de máquina. Observamos muitas inovações contínuas n no espaço do processamento de linguagem natural (PLN). Saudamos também o surgimento e a maturidade contínua dos conjuntos de ferramentas de aprendizado de máquina de ciclo completo, combinando práticas perenes de engenharia com combinações de ferramentas que funcionam bem de maneira iterativa, mostrando que "o aprendizado de máquina também é software". Finalmente, para arquiteturas distribuídas, como microsserviços, vemos grande interesse na malha de dados como uma maneira de efetivamente servir e usar dados analíticos em escala em sistemas distribuídos. À medida que a indústria pensa com mais diligência sobre como os dados devem funcionar nos sistemas modernos, somos encorajadas pela direção geral e pelas perspectivas de abertura nesse cenário, e esperamos ver inovações interessantes no futuro próximo.

Explosão cambriana de Kubernetes & Co.

À medida que o Kubernetes continua a consolidar seu domínio de mercado, o inevitável ecossistema de suporte prospera. Discutimos vários blips relacionados a Kubernetes nos quadrantes de ferramentas, plataformas e técnicas, mostrando o quão difundido esse assunto se tornou. Por exemplo, Lens e k9s simplificam o gerenciamento de cluster, kind ajuda nos testes locais e Gloo oferece um gateway de API alternativo. Hydra é um servidor OAuth otimizado para execução no Kubernetes, e Argo CD usa o gerenciamento de estado desejado nativo do Kubernetes para implementar um servidor de CD. Esses desenvolvimentos indicam que o Kubernetes está perfeitamente preparado para criar um ecossistema de apoio. Oferece recursos críticos, mas com abstrações que geralmente são muito baixo nível ou avançadas para a maioria dos usuários. Assim, o vazio de complexidade é preenchido com ferramentas para facilitar a configuração e o uso do Kubernetes, ou fornecer algo que falta na funcionalidade principal. Enquanto o Kubernetes mantém seu domínio, vemos um rico ecossistema crescendo e se expandindo para aproveitar seus pontos fortes e solucionar suas fraquezas. À medida que esse ecossistema amadurece, esperamos que ele evolua para um novo conjunto de abstrações de nível superior que ofereça os benefícios do Kubernetes sem a confusa gama de opções.

© ThoughtWorks, Inc. Todos os direitos reservados.

O Radar

Técnicas

Adote
1. Aplicando gestão de produto a plataformas internas
2. Infraestrutura como código
3. Micro frontends
4. Pipelines como código
5. Pareamento remoto pragmático
6. Feature toggles mais simples possíveis

Experimente
7. Entrega contínua para aprendizado de máquina (CD4ML)
8. Testes de viés ético
9. GraphQL para agregação de recursos do lado do servidor
10. Micro frontends para aplicativos móveis
11. Times de produto de engenharia de plataforma
12. Políticas de segurança como código
13. Loops de aprendizado semi-supervisionados
14. Transferência de aprendizado para PLN
15. Processos e abordagens "nativamente remotos"
16. Arquitetura de confiança zero (ZTA)

Avalie
17. Malha de dados
18. Identidade descentralizada
19. Definição de pipeline de dados declarativa
20. DeepWalk
21. Gerenciamento de sistemas com estado usando plataformas de orquestração de contêineres
22. Compilações preflight

Evite
23. Lift and shift para nuvem
24. Paridade de funcionalidades na migração de legados
25. Agregação de logs para análise de negócio
26. Branches de longa duração com Gitflow
27. Apenas testes de captura instantânea

Plataformas

Adote
28. .NET Core
29. Istio

Experimente
30. Anka
31. Argo CD
32. Crowdin
33. eBPF
34. Firebase
35. Hot Chocolate
36. Hydra
37. OpenTelemetry
38. Snowflake

Avalie
39. Anthos
40. Apache Pulsar
41. Cosmos
42. Google BigQuery ML
43. JupyterLab
44. Marquez
45. Matomo
46. MeiliSearch
47. Stratos
48. Trillian

Evite
49. Uso excessivo de Node.js

Ferramentas

Adote
50. Cypress
51. Figma

Experimente
52. Dojo
53. DVC
54. Ferramentas de rastreamento de experimentos para aprendizado de máquina
55. Goss
56. Jaeger
57. k9s
58. kind
59. mkcert
60. MURAL
61. Open Policy Agent (OPA)
62. Optimal Workshop
63. Phrase
64. ScoutSuite
65. Ferramentas de teste de regressão visual
66. Visual Studio Live Share

Avalie
67. Apache Superset
68. AsyncAPI
69. ConfigCat
70. Gitpod
71. Gloo
72. Lens
73. Manifold
74. Sizzy
75. Snowpack
76. tfsec

Evite

Linguagens e frameworks

Adote
77. React Hooks
78. Biblioteca de Testes para React
79. Vue.js

Experimente
80. CSS em JS
81. Exposed
82. GraphQL Inspector
83. Karate
84. Koin
85. NestJS
86. PyTorch
87. Rust
88. Sarama
89. SwiftUI

Avalie
90. Clinic.js Bubbleprof
91. Deequ
92. ERNIE
93. MediaPipe
94. Tailwind CSS
95. Tamer
96. Wire
97. XState

Evite
98. Enzyme

Novo Modificado Sem Modificação

TECHNOLOGY RADAR Vol. 22

Técnicas

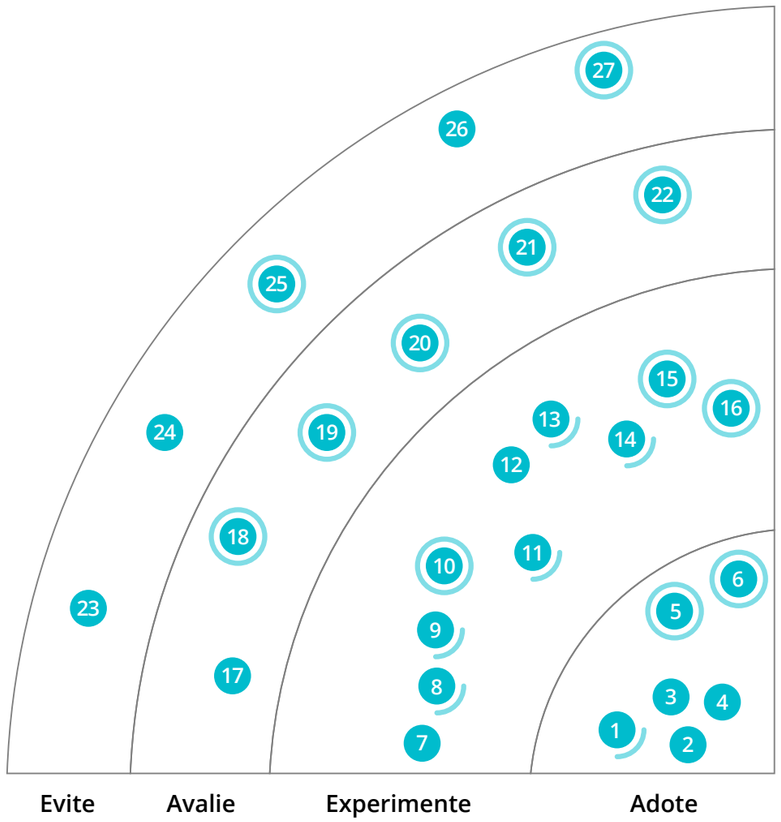
Técnicas

Aplicando gestão de produto a plataformas internas
Adote

Cada vez mais empresas estão construindo plataformas internas para implantar novas soluções digitais de maneira rápida e eficiente. As empresas que obtêm sucesso com essa estratégia estão aplicando gestão de produto a plataformas internas. Isso significa estabelecer empatia com o público consumidor interno (os times de desenvolvimento) e colaborar com eles no design. Gerentes de produto da plataforma criam roteiros e garantem que a plataforma agregue valor aos negócios e aprimore a experiência de desenvolvimento. Infelizmente, também estamos vendo abordagens menos bem-sucedidas, nas quais os times criam uma plataforma vazia, com base em suposições não-verificadas e sem considerar clientes internos. Essas plataformas frequentemente, apesar de táticas internas agressivas, acabam sendo subutilizadas e prejudicam a capacidade de entrega da organização. Como sempre, uma boa gestão de produto tem a ver com a construção de produtos que o público consumidor adora.

Infraestrutura como código
Adote

Embora a infraestrutura como código seja uma técnica relativamente antiga (que destacamos no Radar em 2011), ela se tornou extremamente importante na era moderna da nuvem, em que o ato de configurar a infraestrutura se tornou a aprovação das instruções de configuração para uma plataforma em nuvem. Quando dizemos "como código", queremos dizer que todas as boas práticas que aprendemos no mundo do software devem ser aplicadas à infraestrutura. Uso do controle de origem, adesão ao princípio DRY, modularização, manutenção e uso de testes e implantação automatizados são práticas fundamentais. Aquelas de nós com profundo conhecimento de software e infraestrutura precisam ter empatia e apoiar colegas que ainda não o têm. Dizer "trate a infraestrutura como código" não é suficiente, precisamos garantir que os aprendizados conquistados com muito esforço no mundo do software sejam aplicados de forma consistente em todo o domínio da infraestrutura.



Micro frontends
Adote

Temos visto significativos benefícios de se introduzir microsserviços, que permitem aos times escalar a entrega de serviços mantidos e implantados independentemente. Infelizmente, também vemos muitos times criarem um monólito de frontend – uma grande e confusa aplicação de navegador que fica em cima de serviços de backend –, neutralizando fortemente os benefícios dos microsserviços. Os micro frontends continuam a ganhar popularidade desde que foram introduzidos. Temos visto muitos times adotarem alguma forma dessa

Adote
1. Aplicando gestão de produto a plataformas internas
2. Infraestrutura como código
3. Micro frontends
4. Pipelines como código
5. Pareamento remoto pragmático
6. Feature toggles mais simples possíveis

Experimente
7. Entrega contínua para aprendizado de máquina (CD4ML)
8. Testes de viés ético
9. GraphQL para agregação de recursos do lado do servidor
10. Micro frontends para aplicativos móveis
11. Times de produto de engenharia de plataforma
12. Políticas de segurança como código
13. Loops de aprendizado semi-supervisionados
14. Transferência de aprendizado para PLN
15. Processos e abordagens "nativamente remotos"
16. Arquitetura de confiança zero (ZTA)

Avalie
17. Data mesh
18. Decentralized identity
19. Declarative data pipeline definition
20. DeepWalk
21. Managing stateful systems via container orchestration
22. Preflight builds

Evite
23. Lift and shift para nuvem
24. Paridade de funcionalidades na migração de legados
25. Agregação de logs para análise de negócio
26. Branches de longa duração com Gitflow
27. Apenas testes de captura instantânea

© ThoughtWorks, Inc. All Rights Reserved.

Técnicas

Acreditamos firmemente na programação em pares. Porém, em um momento de domínio do trabalho remoto provocado pela COVID-19, o pareamento exige uma dose saudável de pragmatismo para ser efetivo.

(Pareamento remoto pragmático)

Recomendamos usar feature toggles mais simples possíveis em vez de frameworks de feature toggle desnecessariamente complexos.

(Feature toggles mais simples possíveis)

arquitetura como uma maneira de gerenciar a complexidade de múltiplas pessoas desenvolvedoras e times contribuindo para a mesma experiência de usuário. Em junho deste ano, um dos criadores dessa técnica publicou um artigo introdutório, que serve como referência para micro frontends. Ele mostra como esse estilo pode ser implementado usando vários mecanismos de programação web e constrói um exemplo de aplicação usando React.js. Estamos confiantes de que esse estilo vai crescer em popularidade à medida que grandes organizações tentam dividir o desenvolvimento de UI entre múltiplos times.

Pipelines como código

Adote

A técnica de pipelines como código enfatiza que a configuração dos pipelines de entrega que criam, testam e implantam nossas aplicações ou infraestrutura deve ser tratada como código. Eles devem ser colocados sob controle de origem e modularizados em componentes reutilizáveis com teste e implantação automatizados. À medida que as organizações transitam para times autônomos descentralizados, construindo microsserviços ou micro frontends, a necessidade de práticas de engenharia no gerenciamento de pipelines como código aumenta, para que os times continuem criando e implantando software consistente dentro da organização. Essa necessidade deu origem a modelos e ferramentas de pipeline de entrega que permitem uma maneira padronizada de criar e implantar serviços e aplicações. Essas ferramentas usam pipelines de entrega declarativos das aplicações, adotando um blueprint de pipeline para executar as tarefas subjacentes a vários estágios de um ciclo de entrega — como compilação, teste e implantação —, e eles abstraem os detalhes da implementação. A capacidade de criar, testar e implantar pipelines como código deve ser um dos critérios de avaliação para a escolha de uma ferramenta de CI/CD.

Pareamento remoto pragmático

Adote

Acreditamos firmemente que a programação em pares melhora a qualidade do código, difunde conhecimento entre todo o time e permite a entregar software com mais rapidez. No mundo pós-COVID, no entanto, muitos times de software serão distribuídos ou totalmente remotas e, nessa situação, recomendamos pareamento remoto pragmático: ajustar as práticas de pareamento ao que é possível dadas as ferramentas disponíveis. Considere ferramentas como Visual Studio Live Share para uma colaboração eficiente e de baixa latência. Somente recorra ao compartilhamento de pixels se as duas pessoas participantes residirem em relativa proximidade geográfica e tiverem conexões de Internet de alta largura de banda. Junte pares que estejam em fusos horários semelhantes, em vez de esperar que o pareamento funcione independentemente da localização das pessoas participantes. Se o pareamento não estiver funcionando por razões logísticas, retorne a práticas como programação individual aumentada por meio de revisões de código, colaboração por pull-request (mas atente-se para branches de longa duração com Gitflow) ou sessões de pareamento mais curtas para partes críticas do código. Praticamos pareamento remoto há anos e achamos que ele é eficaz se realizado com uma dose de pragmatismo.

Feature toggles mais simples possíveis

Adote

Infelizmente, feature toggles são menos comuns do que gostaríamos, e muitas vezes vemos pessoas misturando seus tipos e casos de uso. É bastante comum encontrar times que usam plataformas pesadas como LaunchDarkly para implementar feature toggles, incluindo release toggles, para se beneficiar da Integração Contínua, quando tudo o que você precisa são condicionais if/else. Portanto, a menos que você precise de testes A/B ou releases canário, ou transferir a responsabilidade de feature releases a pessoas de negócios, recomendamos que você use feature toggles mais simples possíveis em vez de frameworks de feature toggle desnecessariamente complexos.

Entrega contínua para aprendizado de máquina (CD4ML)

Experimente

Aplicar aprendizado de máquina para tornar serviços e aplicações de negócios inteligentes é mais do que apenas treinar servir modelos. Requer a implementação de ciclos completos e repetíveis de treinamento, testes, implantação, monitoramento e operação dos modelos. Entrega contínua para aprendizado de máquina (CD4ML) é uma técnica que permite ciclos de desenvolvimento de ponta-a-ponta confiáveis, implantação e monitoramento de modelos de aprendizado de máquina. A stack tecnológica subjacente para ativar CD4ML inclui ferramentas para acessar e descobrir dados, controle de versão de artefatos (como dados, modelo e código), pipelines de entrega contínua, provisionamento de ambiente automatizado

© ThoughtWorks, Inc. Todos os direitos reservados.

para várias implantações e experimentos, avaliação e rastreamento de desempenho e modelo, e observabilidade de modelo operacional. As empresas podem escolher seu próprio conjunto de ferramentas, dependendo da stack tecnológica existente. CD4ML enfatiza a automação e a remoção de transferências manuais. CD4ML é a nossa abordagem de escolha para o desenvolvimento de modelos de aprendizado de máquina.

Testes de viés ético

Experimente

No ano passado, vimos uma mudança no interesse em torno de aprendizado de máquina e de redes neurais profundas em particular. Até agora, o desenvolvimento de ferramentas e técnicas foi impulsionado pelo entusiasmo com as notáveis capacidades desses modelos. Atualmente, porém, há uma preocupação crescente de que esses modelos possam causar danos não-intencionais. Por exemplo, um modelo pode ser treinado para tomar decisões de crédito inadvertidamente, simplesmente excluindo pessoas candidatas desfavorecidas. Felizmente, estamos vendo um interesse crescente em testes de viés ético, que ajudarão a apontar decisões potencialmente prejudiciais. Ferramentas como lime, AI Fairness 360 ou What-If Tool podem ajudar a descobrir imprecisões resultantes de grupos sub-representados em dados de treinamento, enquanto ferramentas de visualização como Google Facets ou Facets Dive podem ser usadas para descobrir subgrupos em um corpus de dados de treinamento. Utilizamos lime (*local interpretable model-agnostic explanations*, ou explicações independentes de modelo interpretáveis localmente), além desta técnica, para entender as previsões de qualquer classificador de aprendizado de máquina e o que os classificadores (ou modelos) estão fazendo.

GraphQL para agregação de recursos do lado do servidor

Experimente

Vemos mais e mais ferramentas como Apollo Federation, que pode agregar vários endpoints do GraphQL em um único grafo. No entanto, advertimos contra o uso indevido do GraphQL, especialmente ao transformá-lo em um protocolo de servidor para servidor. Nossa prática é usar GraphQL para agregação de recursos do lado do servidor apenas. Ao usar esse padrão, os microsserviços continuam a expor APIs RESTful bem definidas, enquanto serviços agregados ocultos ou BFF (Backend for Frontends) usam os resolvedores GraphQL como a implementação para combinar recursos de outros serviços. A forma do grafo é orientada por exercícios de modelagem de domínio para garantir que a linguagem onipresente seja limitada a subgrafos quando necessário (no caso de "um microsserviço por bounded context"). Essa técnica simplifica a implementação interna de serviços agregados ou BFFs, incentivando uma boa modelagem de serviços para evitar REST anêmico.

Micro frontends para aplicativos móveis

Experimente

Desde que a introduzimos no Radar em 2016, vimos a adoção generalizada de micro frontends para UIs web. Recentemente, no entanto, temos visto projetos ampliando esse estilo arquitetural para incluir também micro frontends para aplicativos móveis. Quando o aplicativo se torna suficientemente grande e complexo, torna-se necessário distribuir o desenvolvimento entre vários times, introduzindo o desafio de manter a autonomia dos times enquanto integram seu trabalho em um único aplicativo. Embora tenhamos visto times criando seus próprias frameworks para permitir esse estilo de desenvolvimento, estruturas de modularização existentes como Atlas e Beehive também pode simplificar o problema de integrar o desenvolvimento de aplicativos por múltiplos times.

Times de produto de engenharia de plataforma

Experimente

A adoção de nuvem e DevOps, embora aumente a produtividade dos times que agora podem se mover mais rapidamente com dependência reduzida de infraestrutura e times de operações centralizados, também restringiu os times que não possuem as habilidades necessárias para autogerenciar uma stack completa de aplicativos e operações. Algumas organizações enfrentaram esse desafio criando times de produto de engenharia de plataforma. Esses times mantêm uma plataforma interna que permite aos times de entrega implantar e operar sistemas com prazo de entrega reduzido e complexidade de stack. A ênfase aqui está nas ferramentas de autoatendimento e suporte orientadas à API, com os times de entrega ainda responsáveis por dar suporte ao que implementam na plataforma. As organizações que consideram estabelecer um time de plataforma devem ter muito cuidado para não criar acidentalmente um time separado de DevOps, nem devem simplesmente renomear sua estrutura existente de hospedagem e operações como uma plataforma. Se você está se perguntando qual é a melhor configuração para os times de plataforma, usamos os conceitos de topologias de time para dividir os times de plataforma em nossos projetos em times de enablement, times de "plataforma em uma plataforma" plataforma e times com foco no fluxo.

Técnicas

Existe uma preocupação crescente de que alguns modelos de aprendizado de máquina possam causar danos não-intencionais. Felizmente, estamos vendo um interesse crescente em testes de viés ético, que ajudarão a descobrir decisões potencialmente prejudiciais.

(Testes de viés ético)

Micro frontends foram amplamente adotados para UIs web. Agora, também estamos vendo esse estilo arquitetural para dispositivos móveis.

(Micro frontends para aplicativos móveis)

© ThoughtWorks, Inc. All Rights Reserved.

Técnicas

As arquiteturas de confiança zero enfatizam a importância de proteger todos os acessos e comunicações e impor políticas como código baseado no menor privilégio.

(Arquitetura de confiança zero (ZTA))

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.

Loops de aprendizado semi-supervisionados

Experimente

Loops de aprendizado semi-supervisionados são uma classe de fluxos de trabalho iterativos de aprendizado de máquina que aproveitam os relacionamentos encontrados em dados não-rotulados. Essas técnicas podem melhorar os modelos combinando conjuntos de dados rotulados e não-rotulados de várias maneiras. Em outros casos, eles comparam modelos treinados em diferentes subconjuntos de dados. Diferentemente do aprendizado não-supervisionado, em que uma máquina infere classes em dados não-rotulados ou técnicas supervisionadas nas quais o conjunto de treinamento é totalmente rotulado, as técnicas semi-supervisionadas aproveitam um pequeno conjunto de dados rotulados e um conjunto muito maior de dados não-rotulados. O aprendizado semi-supervisionado também está intimamente relacionado às técnicas de aprendizado ativo, nas quais um ser humano é direcionado para rotular seletivamente pontos de dados ambíguos. Como as pessoas especialistas que podem rotular dados com precisão são um recurso escasso e a rotulagem costuma ser a atividade que consome mais tempo no fluxo de trabalho de aprendizado de máquina, as técnicas semi-supervisionadas reduzem o custo do treinamento e tornam o aprendizado de máquina viável para uma nova classe de usuários. Também estamos vendo a aplicação de técnicas pouco supervisionadas, em que os dados rotulados por máquina são usados, mas são menos confiáveis do que os dados rotulados por humanos.

Transferência de aprendizado para PLN

Experimente

Nós havíamos incluído essa técnica em Avalie anteriormente. As inovações no cenário do PLN continuam em um ótimo ritmo, e conseguimos aproveitar essas inovações em nossos projetos graças à onipresente transferência de aprendizado para PLN. As pontuações do benchmark GLUE (um conjunto de tarefas de compreensão de linguagem) tiveram um progresso dramático nos últimos dois anos, com pontuações médias passando de 70,0 no lançamento para líderes ultrapassando 90,0 em abril de 2020. Muitos de nossos projetos no domínio PLN são capazes de atingir progressos significativos, iniciando a partir de modelos pré-treinados do ELMo, BERT e ERNIE, entre outros, e depois ajustando-os com base nas necessidades do projeto.

Processos e abordagens “nativamente remotos”

Experimente

Os times distribuídos podem ter diferentes formatos e configurações. Os times de entrega com configuração 100% co-localizada em um único local, no entanto, tornaram-se exceção para nós. A maioria de nossos times se distribui em várias localizações ou tem pelo menos alguns membros trabalhando remotamente. Portanto, adotar processos e abordagens “nativamente remotos” por padrão pode ajudar significativamente no fluxo e na eficácia gerais do time. Isso começa com a garantia de que todos os membros tenham acesso aos sistemas remotos necessários. Além disso, usar ferramentas como Visual Studio Live Share, MURAL ou Jamboard transforma workshops online e pareamento remoto em rotinas, em vez de exceções ineficazes. Mas o “nativamente remoto” vai além de levar e substituir práticas co-localizadas para o mundo digital: adotando comunicação mais assíncrona, ainda mais disciplina em torno da documentação das decisões, e reuniões com “todo mundo sempre remoto” são outras abordagens praticadas por nossos times por padrão para otimizar a fluidez independente da localização.

Arquitetura de confiança zero (ZTA)

Experimente

Atualmente, o panorama tecnológico das organizações é cada vez mais complexo, com ativos — dados, funções, infraestrutura e usuários — espalhados pelos limites de segurança, como hosts locais, vários provedores de nuvem e uma variedade de fornecedores de SaaS. Isso exige uma mudança de paradigma no planejamento da segurança corporativa e na arquitetura

© ThoughtWorks, Inc. Todos os direitos reservados.

dos sistemas, passando do gerenciamento estático e de mudanças lentas de políticas de segurança, baseado em zonas de confiança e configurações de rede, para a aplicação dinâmica e refinada das políticas de segurança baseadas em privilégios de acesso temporais.

A arquitetura de confiança zero (*zero-trust architecture* ou ZTA) é a estratégia e a jornada de uma organização para implementar princípios de segurança de confiança zero para todos os seus ativos — como dispositivos, infraestrutura, serviços, dados e usuários — e inclui práticas de implementação, como garantia de acesso e comunicações independentemente da localização da rede, aplicação de políticas como código baseado no menor privilégio e o mais granular possível, e monitoramento contínuo e mitigação automatizada de ameaças. Nosso Radar reflete muitas das técnicas habilitadoras, como como política de segurança como código, sidecars para segurança de endpoint e BeyondCorp. Se você estiver migrando para ZTA, consulte a publicação do NIST sobre ZTA para saber mais sobre os princípios, componentes de tecnologia habilitadores e padrões de migração, bem como a publicação do Google no BeyondProd.

Malha de dados

Avalie

A malha de dados é um paradigma arquitetural e organizacional que desafia a antiga suposição de que devemos centralizar grandes volumes de dados analíticos para usá-los, manter todos os dados em um só lugar ou gerenciá-los por meio de um time de dados centralizado para entregar valor. A noção de malha de dados afirma que, para que o big data incentive a inovação, sua propriedade deve ser compartilhada entre as partes proprietárias dos dados do domínio responsáveis pelo fornecimento de dados como produtos (com o suporte de uma plataforma de dados de autoatendimento para abstrair a complexidade técnica envolvida no fornecimento de produtos de dados). Também deve-se adotar uma nova forma de governança compartilhada por meio da automação, para permitir a interoperabilidade de produtos de dados orientados ao domínio. A descentralização, juntamente com a interoperabilidade e o foco na experiência de quem consome os dados, são essenciais para a democratização da inovação no uso de dados.

Se sua organização possui um grande número de domínios com vários sistemas e times gerando dados, ou um conjunto diversificado de casos de uso e padrões de acesso controlados por dados, sugerimos que você avalie a malha de dados. A implementação da malha de dados requer investimento na construção de uma plataforma de dados de autoatendimento e na adoção de uma mudança organizacional para que os domínios assumam a propriedade de seus produtos de dados a longo prazo, bem como uma estrutura de incentivos que recompense domínios que atendem e utilizam dados como produto.

Identidade descentralizada

Avalie

Desde o nascimento da Internet, o cenário tecnológico experimentou uma evolução acelerada em direção à descentralização. Embora protocolos como HTTP e padrões de arquitetura como microsserviços ou malha de dados permitam implementações descentralizadas, o gerenciamento de identidades permanece centralizado. O surgimento da tecnologia de ledger distribuído (DLT), no entanto, habilita a possibilidade do conceito de identidade descentralizada. Em um sistema de identidade descentralizada, as entidades – unidades identificáveis, como pessoas, organizações e outras coisas – são livres para usar qualquer root compartilhada de confiança. Por outro lado, os sistemas convencionais de gerenciamento de identidades são baseados em autoridades e registros centralizados, como serviços de diretório corporativo, autoridades de certificação ou registros de nomes de domínio.

O desenvolvimento de identificadores descentralizados – identificadores globalmente únicos, persistentes e auto-soberanos que são criptograficamente verificáveis – é um dos principais padrões de habilitação. Embora as implementações em escala de identificadores descentralizados ainda sejam raras, estamos otimistas com a premissa desse movimento e começamos a usar o conceito em nossas arquiteturas. Para os últimos experimentos e colaborações do setor, consulte a Decentralized Identity Foundation.

Definição de pipeline de dados declarativa

Avalie

Muitos pipelines de dados são definidos em um script grande, mais ou menos imperativo, escrito em Python ou Scala. O script contém a lógica das etapas individuais, bem como o código que as une. Quando confrontadas com uma situação semelhante nos testes Selenium, as pessoas desenvolvedoras descobriram o padrão Objeto de Página e, posteriormente, muitos frameworks de desenvolvimento orientado a comportamento (BDD) implementaram uma divisão entre as definições de etapas e sua composição. Agora, alguns times estão tentando trazer o mesmo pensamento para a engenharia de dados. Uma definição de pipeline de dados declarativa separada, talvez escrita em YAML, contém apenas

© ThoughtWorks, Inc. All Rights Reserved.

Técnicas

A base fundamental da BitCoin — tecnologia de ledger distribuído (DLT) — é permitir o surgimento de identidades descentralizadas.

(Identidade descentralizada)

Técnicas

Ao trabalhar em conjuntos de dados representados como grafos, um dos principais problemas é extrair recursos do grafo. É aqui que o DeepWalk pode ajudar.

(DeepWalk)

a declaração e a sequência de etapas. Ela indica conjuntos de dados de entrada e saída, mas refere-se a scripts se e quando uma lógica mais complexa for necessária. Com o A La Mode, estamos vendo a primeira ferramenta de código aberto a aparecer nesse espaço.

DeepWalk
Avalie

DeepWalk é um algoritmo que ajuda a aplicar aprendizado de máquina em grafos. Ao trabalhar em conjuntos de dados representados como grafos, um dos principais problemas é extrair recursos deles. É aqui que o DeepWalk pode ajudar. Ele usa o SkipGram para construir incorporações de nós, visualizando o grafo como uma linguagem em que cada nó é uma palavra única e percursos aleatórios de comprimento finito no grafo constituem uma sentença. Essas incorporações podem ser usadas por vários modelos de ML. O DeepWalk é uma das técnicas que estamos testando em alguns de nossos projetos nos quais precisamos aplicar aprendizado de máquina em grafos.

Gerenciamento de sistemas com estado usando plataformas de orquestração de contêineres
Avalie

Recomendamos cautela no gerenciamento de sistemas com estado usando plataformas de orquestração de contêineres como o Kubernetes. Alguns bancos de dados não são construídos com suporte nativo para orquestração — eles não esperam que um agendador os mate e realoque para outro host. Criar um serviço altamente disponível em cima de tais bancos de dados não é trivial e ainda recomendamos executá-los em hosts bare metal ou em uma máquina virtual (VM), em vez de forçar o ajuste em uma plataforma de orquestração de contêiner.

Compilações preflight
Avalie

Embora defendamos fortemente a integração contínua em vez de Gitflow, sabemos que executar commits diretamente no trunk e integração contínua em uma master branch pode ser ineficaz se o time for muito grande, se as construções forem lentas ou instáveis ou se o time não tiver disciplina para executar o conjunto de testes completo localmente. Nessa situação, uma construção vermelha pode bloquear várias pessoas desenvolvedoras ou pares. Em vez de corrigir a causa raiz subjacente — construções lentas, a incapacidade de executar testes localmente ou arquiteturas monolíticas que exigem muitas pessoas trabalhando na mesma área — os times geralmente confiam em feature branches para contornar esses problemas. Nós desencorajamos os feature branches, pois eles podem exigir um esforço significativo para resolver conflitos de merge e introduzem ciclos de feedback mais longos e possíveis erros durante a resolução de conflitos. Em vez disso, propomos usar compilações preflight como alternativa: são compilações baseadas em pull request para "microbranches" que permanecem apenas pelo período de execução do pipeline, com o branch aberto para cada commit. Para ajudar a automatizar esse fluxo de trabalho, encontramos bots como Bors, que automatiza o merge para controlar e excluir branches caso a construção da minibranch tenha êxito. Estamos avaliando esse fluxo, e você também deve; mas não use isso para resolver o problema errado, pois pode levar ao uso indevido de branches e causar mais danos do que benefícios.

Lift and shift para nuvem
Evite

É bastante curioso que, após mais de uma década de experiência com migração para nuvem na indústria, ainda sentimos que é necessário chamar atenção para a prática de lift and shift para nuvem ; na qual a nuvem é vista simplesmente como uma solução de hospedagem, e a arquitetura existente, práticas de segurança e modelos operacionais de TI são simplesmente replicados na nuvem. Isso não cumpre as promessas de agilidade e inovação digital da nuvem. Uma migração para nuvem requer uma mudança intencional em vários eixos em direção a um estado nativo da nuvem. Dependendo das circunstâncias únicas da migração, cada organização pode acabar em algum lugar do espectro: desde lift and shift até estruturas nativas de nuvem. A arquitetura de sistemas, por exemplo, é um dos pilares da agilidade na entrega e geralmente requer mudanças. A tentação de simplesmente fazer lift and shift dos sistemas existentes como contêineres para a nuvem pode ser forte. Embora essa tática possa acelerar a migração para a nuvem, ela deixa a desejar na criação de agilidade e entrega de funcionalidades e valor. A segurança corporativa na nuvem é fundamentalmente diferente da segurança tradicional baseada em perímetro por meio de firewalls e zoneamento, e exige uma jornada em direção à arquitetura de confiança zero. O modelo operacional de TI também precisa ser reformulado para fornecer serviços de nuvem com segurança por meio de plataformas automatizadas de autoatendimento e para capacitar os times a assumirem mais responsabilidades operacionais e ganharem autonomia. Por último, mas não menos importante, as organizações devem criar uma base para permitir mudanças contínuas, como a criação de pipelines com testes contínuos

© ThoughtWorks, Inc. Todos os direitos reservados.

de aplicações e infraestrutura como parte da migração. Isso ajudará o processo de migração, resultará em um sistema mais robusto e bem estruturado e dará às organizações uma maneira de continuar evoluindo e melhorando seus sistemas.

Paridade de funcionalidades na migração de legados

Evite

Estamos descobrindo que mais e mais empresas precisam substituir sistemas legados envelhecidos para acompanhar as demandas de clientes (tanto internas quanto externas). Um antipadrão que continuamos vendo é a paridade de funcionalidades na 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 incham com o tempo, com muitos recursos que não são 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 para entender o que os usuários precisam atualmente e priorizar essas necessidades de acordo com resultados de negócios e métricas — o que é muitas vezes mais fácil de falar do que de fazer. Isso significa conduzir uma pesquisa de usuário e aplicar práticas modernas de desenvolvimento de produto em vez de simplesmente substituir as práticas existentes.

Agregação de logs para análise de negócio

Evite

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

Branches de longa duração com Gitflow

Evite

Há cinco anos, destacamos os problemas com branches de longa duração com Gitflow. Essencialmente, branches de longa duração são o oposto da integração contínua de todas as alterações no código-fonte e, em nossa experiência, a integração contínua é a melhor abordagem para a maioria dos tipos de desenvolvimento de software. Mais tarde, estendemos nossa cautela ao próprio Gitflow, porque vimos equipes usando-o quase exclusivamente com branches de longa duração. Hoje, ainda vemos times com configurações nas quais a entrega contínua de sistemas baseados na web é o objetivo declarado sendo atraídos para branches de longa duração. Por isso, ficamos felizes com o fato de o autor do Gitflow ter adicionado uma nota ao seu artigo original, explicando que o Gitflow não se destinava a esses casos de uso.

Apenas testes de captura instantânea

Evite

O valor do teste de captura instantânea é inegável ao se trabalhar com sistemas legados, garantindo que o sistema continue funcionando e o código legado não quebre. No entanto, estamos vendo a prática comum e bastante prejudicial de usar apenas testes de captura instantânea como mecanismo de teste principal. Os testes de captura instantânea validam o resultado exato gerado no DOM por um componente, e não o comportamento do componente; portanto, podem ser fracos e não confiáveis e promover a má prática de "apenas excluir a captura instantânea e regenerá-la". Em vez disso, você deve testar a lógica e o comportamento dos componentes emulando o que os usuários fariam. Essa mentalidade é incentivada pelas ferramentas da família Testing Library.

Técnicas

Os logs destinados à observabilidade técnica geralmente são inadequados para inferir um entendimento profundo de clientes.

(Agregação de logs para análise de negócio)

O teste de captura instantânea é inegavelmente útil ao trabalhar com sistemas legados. Mas não deve ser o mecanismo de teste principal para esses sistemas.

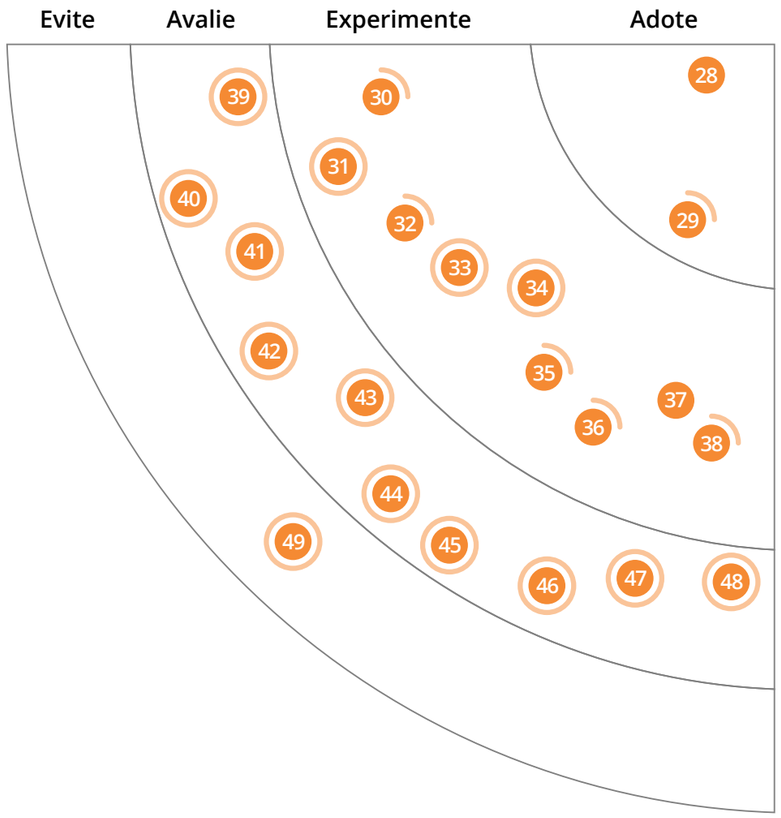
(Apenas testes de captura instantânea)

© ThoughtWorks, Inc. All Rights Reserved.

TECHNOLOGY RADAR Vol. 22

Plataformas

Plataformas



.NET Core
Adote

Anteriormente, tínhamos incluído o .NET Core em Adote, indicando que ele havia se tornado nosso padrão para projetos .NET. Mas sentimos que vale a pena chamar novamente a atenção para ele. Com o lançamento do .NET Core 3.x no ano passado, a maior parte dos recursos do .NET Framework agora foi portada para o .NET Core. Com o anúncio de que o .NET Framework está em sua última versão, a Microsoft reforçou a visão de que o .NET Core é o futuro do .NET. A Microsoft trabalhou bastante para tornar o .NET Core compatível com contêineres. A maioria dos nossos projetos baseados no .NET Core tem como alvo o Linux, e geralmente são implantados como contêineres. A próxima versão do .NET 5 parece promissora e estamos otimistas.

Istio
Adote

Se você estiver construindo e operando uma arquitetura de microsserviços em escala e abraçou o Kubernetes, adotar a malha de serviços para gerenciar todos os aspectos transversais da execução da arquitetura é uma posição padrão. Entre as várias implementações de malha de serviço, Istio conquistou adoção majoritária. Possui um rico conjunto de funcionalidades, incluindo descoberta de serviço, gerenciamento de tráfego, segurança de serviço-a-serviço e origem-a-serviço, observabilidade (incluindo telemetria e rastreamento distribuído), releases contínuos e resiliência. Sua experiência de uso foi aprimorada em seus últimos releases, devido à sua facilidade de instalação e arquitetura do painel de controle. O Istio reduziu o nível de dificuldade de implementação de microsserviços em larga escala com qualidade operacional para muitas de nossas clientes, ao mesmo tempo em que admitimos que operar suas próprias instâncias do Istio e Kubernetes requer conhecimento e recursos internos adequados e não é para quem não quer fortes emoções.

Anka
Experimente

Anka é um conjunto de ferramentas para criar, gerenciar, distribuir, criar e testar ambientes virtuais reproduzíveis do macOS para iOS e macOS. Ele traz uma experiência semelhante ao Docker para ambientes macOS: início instantâneo, CLI para gerenciar máquinas virtuais, e registro para versão e identificação de máquinas virtuais para distribuição. Usamos Anka para criar uma nuvem privada do macOS para uma cliente. Vale a pena considerar

Adote
28. .NET Core
29. Istio

Experimente
30. Anka
31. Argo CD
32. Crowdin
33. eBPF
34. Firebase
35. Hot Chocolate
36. Hydra
37. OpenTelemetry
38. Snowflake

Avalie
39. Anthos
40. Apache Pulsar
41. Cosmos
42. Google BigQuery ML
43. JupyterLab
44. Marquez
45. Matomo
46. MeiliSearch
47. Stratos
48. Trillian

Evite
49. Uso excessivo de Node.js

© ThoughtWorks, Inc. All Rights Reserved.

Plataformas

Embora essa tecnologia não seja nova, agora está ganhando força com o crescente uso de microsserviços implantados como contêineres orquestrados.

(eBPF)

A nítida separação de identidade do restante da estrutura do OAuth2 facilita a integração do Hydra com um ecossistema de autenticação existente.

(Hydra)

essa ferramenta para virtualizar ambientes iOS e macOS.

Argo CD
Experimente

Sem julgar a técnica de GitOps, gostaríamos de falar sobre o Argo CD no escopo de implantação e monitoramento de aplicativos no Kubernetes. Com base em sua capacidade de automatizar a implantação do estado desejado da aplicação nos ambientes de destino especificados no Kubernetes e em nossa boa experiência na solução de problemas de implantações com falha, verificação de logs e monitoramento do status de implantação, recomendamos que você experimente o Argo CD. Você pode até mesmo visualizar graficamente o que está acontecendo no cluster, como uma mudança é propagada e como os pods são criados e destruídos em tempo real.

Crowdin
Experimente

A maioria dos projetos com suporte multi-idiomas começa com os times de desenvolvimento criando funcionalidades em um idioma e gerenciando o restante por meio da tradução offline por e-mails e planilhas. Embora essa configuração simples funcione, as coisas podem sair rapidamente do controle. Pode ser necessário continuar respondendo as mesmas perguntas para diferentes pessoas tradutoras, desgastando a colaboração entre pessoas tradutoras, revisoras e time de desenvolvimento. Crowdin é uma das poucas plataformas que ajudam a otimizar o fluxo de trabalho de localização do seu projeto. Com o Crowdin, o time de desenvolvimento pode continuar criando funcionalidades, enquanto a plataforma otimiza textos que precisam de tradução para um fluxo de trabalho online. Gostamos do fato de Crowdin incentivar os times a incorporar traduções de forma contínua e incremental, em vez de gerenciá-las em grandes lotes no final.

eBPF
Experimente

Há vários anos, o kernel do Linux inclui a máquina virtual estendida do Berkeley Packet Filter (eBPF) e fornece a capacidade de anexar filtros eBPF a soquetes específicos. Mas o extended BPF vai muito além da filtragem de pacotes e permite que scripts personalizados sejam acionados em vários pontos do kernel com muito pouca sobrecarga. Embora essa tecnologia não seja nova, agora está amadurecendo com o uso crescente de microsserviços implantados como contêineres orquestrados. As comunicações serviço-a-serviço podem ser complexas nesses sistemas, dificultando a correlação de problemas de latência ou desempenho com uma chamada de API. Estamos vendo hoje ferramentas lançadas com scripts eBPF pré-escritos para coletar e visualizar o tráfego de pacotes ou gerar relatórios sobre a utilização da CPU. Com o crescimento do Kubernetes, estamos vendo uma nova geração de aplicação e instrumentação de segurança baseada em scripts eBPF que ajudam a dominar a complexidade de uma implantação ampla de microsserviços.

Firebase
Experimente

O Firebase do Google passou por uma evolução significativa desde que o mencionamos como parte de uma arquitetura sem servidor em 2016. O Firebase é uma plataforma abrangente para a criação de aplicativos móveis e web de uma maneira que é suportada pela infraestrutura escalável subjacente do Google. Gostamos particularmente do Firebase App Distribution, que facilita a publicação de versões de teste de um aplicativo por meio de um pipeline de CD, e do Firebase Remote Config, que permite que as alterações na configuração sejam dinamicamente enviadas aos aplicativos sem a necessidade de republicá-los.

Hot Chocolate
Experimente

O ecossistema e a comunidade GraphQL continuam crescendo. Hot Chocolate é um servidor GraphQL para .NET (Core e Classic). Permite criar e hospedar esquemas e, em seguida, realizar consultas neles usando os mesmos componentes básicos do GraphQL — carregador de dados, resolvedor, esquema, operações e tipos. A equipe por trás do Hot Chocolate adicionou recentemente a costura de esquema, que permite que um único ponto de entrada faça consultas em vários esquemas agregados em diferentes locais. Apesar do potencial uso indevido dessa abordagem, nossos times estão satisfeitos com o Hot Chocolate — ele é bem documentado e nos permite entregar valor para clientes rapidamente.

Hydra
Experimente

Nem todo mundo precisa de uma solução OAuth2 auto-hospedada, mas se você precisar, dê uma olhada no Hydra — um servidor OAuth2 de código aberto totalmente compatível e provedor de conexão OpenID. O Hydra possui suporte de armazenamento na memória para

© ThoughtWorks, Inc. Todos os direitos reservados.

desenvolvimento e um banco de dados relacional (PostgreSQL) para casos de uso de produção. O Hydra, como tal, é sem estado e fácil de escalar horizontalmente em plataformas como Kubernetes. Dependendo do seu requisito de desempenho, pode ser necessário ajustar o número de instâncias do banco de dados enquanto dimensiona as instâncias do Hydra. E como ele não fornece nenhuma solução de gerenciamento de identidade pronta para o uso, você pode integrar qualquer tipo de gerenciamento de identidade ao Hydra por meio de uma API limpa. Essa nítida separação de identidade do restante do framework OAuth2 facilita a integração do Hydra com um ecossistema de autenticação existente.

OpenTelemetry
Experimente

OpenTelemetry é um projeto de observabilidade de código aberto que mescla OpenTracing e OpenCensus. O projeto OpenTelemetry inclui especificação, bibliotecas, agentes e outros componentes necessários para capturar telemetria dos serviços para melhor observá-los, gerenciar e depurá-los. Ele abrange os três pilares da observabilidade — rastreamento distribuído, métricas e log (atualmente em beta) — e sua especificação conecta essas três partes por meio de correlações. Portanto, você pode usar metrics para identificar um problema, localizar traces correspondentes para descobrir onde o problema ocorreu e, finalmente, estudar os logs correspondentes para encontrar a causa raiz exata. Os componentes OpenTelemetry podem ser conectados a sistemas de observabilidade de backend como Prometheus e Jaeger, entre outros. A formação do OpenTracing é um passo positivo em direção à convergência da padronização e à simplificação das ferramentas.

Snowflake
Experimente

Snowflake provou ser uma solução robusta de armazenamento, warehouse ou data lake SaaS para muitas de nossas clientes. Possui uma arquitetura superior para dimensionar armazenamento, computação e serviços para carregar, descarregar e usar dados. Também é muito flexível: suporta armazenamento de dados estruturados, semiestruturados e não-estruturados; fornece uma lista crescente de conectores para diferentes padrões de acesso, como Spark para ciência de dados e SQL para análise; e é executado em vários provedores de nuvem. Nosso conselho para muitas de nossas clientes é usar serviços gerenciados para sua tecnologia de utilidade, como armazenamento de big data; no entanto, se o risco e os regulamentos proibirem o uso de serviços gerenciados, Snowflake é um bom candidato para empresas com grandes volumes de dados e cargas de trabalho de processamento pesado. Embora tenhamos tido sucesso usando Snowflake em nossos engajamentos de médio porte, ainda precisamos experimentá-lo em grandes ecossistemas nos quais os dados precisam pertencer a diversos segmentos da organização.

Anthos
Avalie

Vemos uma mudança nos planos de migração acidental de nuvem híbrida ou de toda a propriedade para estratégias intencionais e sofisticadas de nuvem híbrida, polycloud ou portabilidade, nas quais as organizações aplicam princípios multidimensionais para estabelecer e executar sua estratégia de nuvem: onde hospedar seus vários dados e ativos funcionais com base em perfis de risco, capacidade de controle e desempenho; como utilizar seus investimentos em infraestrutura no local e reduzir o custo das operações; e como tirar proveito de vários provedores de nuvem e seus serviços diferenciados exclusivos, sem criar complexidade e atrito para os usuários que constroem e operam aplicativos.

Anthos é a resposta do Google para habilitar estratégias híbridas e multicloud, fornecendo um plano de gerenciamento e controle de alto nível sobre um conjunto de tecnologias de código aberto, como GKE, Service Mesh e um gerenciamento de configurações baseado em Git. Ele permite executar cargas de trabalho portáteis e outros ativos em diferentes ambientes de hospedagem, incluindo o Google Cloud e o hardware local. Enquanto outros provedores de nuvem têm ofertas semelhantes, Anthos pretende ir além de uma nuvem híbrida, funcionando como um facilitador de nuvem portátil usando componentes de código aberto, mas isso ainda está por acontecer. Temos visto crescente interesse no Anthos. Embora a abordagem do Google para nuvem híbrida gerenciada pareça promissora, ela não é mágica, e requer alterações nos recursos existentes na nuvem e no local. Nosso conselho para clientes que consideram usar o Anthos é fazer escolhas ponderadas entre a seleção de serviços do ecossistema do Google Cloud e outras opções, para manter o nível certo de neutralidade e controle.

Apache Pulsar
Avalie

Apache Pulsar é uma plataforma de mensagens/streaming pub-sub de código aberto, competindo em um espaço semelhante ao do Apache Kafka. Ele fornece as funcionalidades esperadas — como entrega de mensagens assíncronas e síncronas com baixa latência e armazenamento persistente e escalável de mensagens —, bem como várias bibliotecas de clientes. O que nos entusiasmou a

Plataformas

O projeto OpenTelemetry inclui especificações, bibliotecas, agentes e outros componentes necessários para capturar telemetria dos serviços, para melhor observá-los, gerenciá-los e depurá-los.

(OpenTelemetry)

Anthos é a resposta do Google para ativar estratégias híbridas e multicloud. Ele permite que você execute cargas de trabalho portáteis em diferentes ambientes de hospedagem, incluindo Google Cloud e hardware local.

(Anthos)

© ThoughtWorks, Inc. All Rights Reserved.

Plataformas

O BigQuery ML reduz o nível de dificuldade em usar ML para previsões e recomendações, principalmente para explorações rápidas.

(Google BigQuery ML)

avaliar o Pulsar foi sua facilidade de escalabilidade, principalmente em grandes organizações com vários segmentos de usuários. Pulsar suporta nativamente multitenancy, replicação geográfica, controle de acesso baseado em papéis e segregação de cobrança. Também estamos considerando Pulsar para resolver o problema de um log sem fim de mensagens para nossos sistemas de dados em larga escala, onde os eventos devem persistir indefinidamente e assinantes podem começar a consumir mensagens retrospectivamente. Isso é suportado por um modelo de armazenamento em camadas. Embora Pulsar seja uma plataforma promissora para grandes organizações, há espaço para melhorias. Sua instalação atual requer a administração de ZooKeeper e BookKeeper, entre outras peças de tecnologia. Esperamos que, com sua crescente adoção, os usuários possam contar em breve com um suporte mais amplo da comunidade.

Cosmos
Avalie

O desempenho da tecnologia de blockchain melhorou bastante desde que a avaliamos inicialmente no Radar. No entanto, ainda não existe uma única blockchain que possa atingir uma taxa de transferência "no nível da Internet". À medida que várias plataformas de blockchain se desenvolvem, estamos vendo novos silos de dados e valor. É por isso que a tecnologia de cadeia cruzada sempre foi um tópico importante na comunidade blockchain: o futuro da blockchain pode ser uma rede de blockchains paralelas independentes. Essa também é a visão do Cosmos. O Cosmos libera Tendermint e CosmosSDK para permitir que as pessoas desenvolvedoras personalizem blockchains independentes. Essas blockchains paralelas podem trocar valor por meio do protocolo Inter-Blockchain Communication (IBC) e Peg-Zones. Nossos times tiveram ótimas experiências com o CosmosSDK e o protocolo IBC está amadurecendo. Essa arquitetura pode resolver problemas de interoperabilidade e escalabilidade de blockchain.

Google BigQuery ML
Avalie

Frequentemente, o treinamento e a previsão de resultados dos modelos de aprendizado de máquina exigem código para levar os dados ao modelo. Google BigQuery ML inverte essa lógica trazendo o modelo para os dados. Google BigQuery é um armazém de dados projetado para atender a consultas em larga escala usando SQL, para casos de uso analíticos. O Google BigQuery ML estende essa função e sua interface SQL para criar, treinar e avaliar modelos de aprendizado de máquina usando seus conjuntos de dados e, eventualmente, executar previsões de modelo para criar novos conjuntos de dados do BigQuery. Ele suporta um conjunto limitado de modelos prontos para uso, como regressão linear para previsão ou regressão binária e multiclasse para classificação. Ele também suporta, com funcionalidade limitada, a importação de modelos TensorFlow previamente treinados. Embora o BigQuery ML e sua abordagem baseada em SQL reduzam a dificuldade de uso do aprendizado de máquina para fazer previsões e recomendações, particularmente para explorações rápidas, isso vem com um desafio: comprometer outros aspectos do treinamento de modelos, como testes de viés ético, explicabilidade e entrega contínua para aprendizado de máquina.

JupyterLab
Avalie

JupyterLab é a interface de usuário web de última geração do projeto Jupyter. Se você usa os notebooks Jupyter, vale a pena tentar o JupyterLab. Ele fornece um ambiente interativo para notebooks, código e dados Jupyter. Nós o vemos como uma evolução do Jupyter Notebook: ele fornece uma experiência melhor, ampliando seus recursos originais, permitindo que código, visualização e documentação existam em um só lugar.

Marquez
Avalie

Marquez é um projeto de código aberto relativamente jovem, para coletar e fornecer informações de metadados sobre um ecossistema de dados. Ele representa um modelo de dados simples para capturar metadados, como linhagem, fazer upstream e downstream de trabalhos de processamento de dados e seu status, e conta com um conjunto flexível de tags para capturar atributos dos conjuntos de dados. Ele fornece uma API RESTful simples para gerenciar os metadados, o que facilita a integração do Marquez a outros conjuntos de ferramentas no ecossistema de dados. Usamos Marquez como ponto de partida e estendemos seu uso com facilidade para atender às nossas necessidades, como impor políticas de segurança e alterações na linguagem do domínio. Se você está procurando uma ferramenta pequena e simples para iniciar o armazenamento e a visualização de suas tarefas de processamento de dados e conjuntos de dados, Marquez é um bom ponto de partida.

© ThoughtWorks, Inc. Todos os direitos reservados.

Matomo
Avalie

Matomo (anteriormente Piwik) é uma plataforma de web analytics de código aberto que fornece controle total sobre seus dados. Você pode auto-hospedar o Matomo e proteger seus dados de web analytics de terceiros. O Matomo também facilita a integração de dados de web analytics com sua plataforma de dados interna e permite criar modelos de uso adaptados às suas necessidades.

MeiliSearch
Avalie

MeiliSearch é um mecanismo de pesquisa de texto rápido, fácil de usar e fácil de implantar. Ao longo dos anos, o Elasticsearch se tornou a escolha popular para pesquisas de texto escaláveis. No entanto, se você não possui um volume de dados que justifique uma solução distribuída, mas ainda deseja fornecer um mecanismo de pesquisa rápido e tolerante a erros de digitação, recomendamos a avaliação do MeiliSearch.

Stratos
Avalie

Ultraleap (anteriormente Leap Motion) tem liderado o espaço de Realidade Estendida (XR) há algum tempo, criando um hardware notável de rastreamento de mãos que permite que as mãos do usuário saltem para a realidade virtual. Stratos é a plataforma de sensores táteis e software subjacente da Ultraleap, e pode usar ultrassom direcionado para criar feedback tátil no ar. Um caso de uso é a resposta ao gesto da mão de um motorista para mudar o ar-condicionado no carro e o feedback tátil como parte da interface. Estamos otimistas para ver a evolução dessa tecnologia e como tecnologistas usarão a criatividade para incorporá-la em seus casos de uso.

Trillian
Avalie

Trillian é um armazenamento de dados centralizado e criptograficamente verificável. Para ambientes descentralizados e sem confiança, você pode usar ledgers distribuídos baseados em blockchain. Para ambientes corporativos, no entanto, onde protocolos de consenso que fazem alto uso de CPU o custo é injustificado, recomendamos que você experimente Trillian.

Uso excessivo de Node.js
Evite

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

Plataformas

MeiliSearch é um mecanismo de pesquisa de texto rápido, fácil de usar e fácil de implantar. É ideal caso você não possua um volume de dados que justifique uma solução distribuída, mas ainda deseja fornecer um mecanismo de pesquisa rápido e tolerante a erros de digitação.

(MeiliSearch)

Stratos é plataforma de sensores táteis e software subjacente que alimenta o XR pioneer Ultraleap (anteriormente Leap Motion).

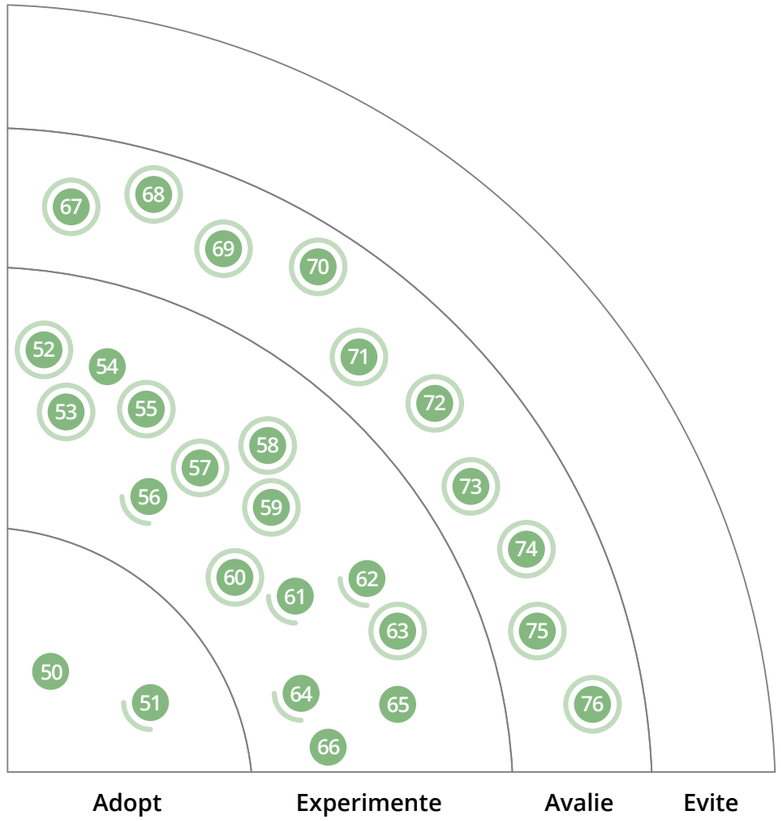
(Stratos)

© ThoughtWorks, Inc. All Rights Reserved.

TECHNOLOGY RADAR Vol. 22

Ferramentas

Ferramentas



Cypress
Adote

Cypress ainda é um favorito entre nossos times nos quais as pessoas desenvolvedoras gerenciam os testes de ponta-a-ponta, como parte de uma pirâmide de testes saudável, é claro. Decidimos destacar novamente neste Radar porque as versões recentes do Cypress adicionaram suporte ao Firefox, e sugerimos fortemente testar em múltiplos navegadores. O domínio dos navegadores Chrome e baseados no Chromium levou os times a uma tendência preocupante de aparentemente testar apenas no Chrome, o que pode levar a surpresas desagradáveis.

Figma
Adote

Figma demonstrou ser a ferramenta de escolha para design colaborativo, não apenas para designers, mas também para times multidisciplinares. Ele permite que pessoas desenvolvedoras e de outros papéis visualizem e comentem projetos pelo navegador sem a versão desktop. Em comparação com seus concorrentes (Invision ou Sketch, por exemplo), que exigem mais de uma ferramenta para versionamento, colaboração e compartilhamento de design, o Figma reúne todas essas funcionalidades em uma ferramenta que facilita a descoberta de novas ideias em conjunto pelos times. Nossos times consideram o Figma muito útil, especialmente para habilitação e facilitação remotas do trabalho de design. Além de seus recursos de design e colaboração em tempo real, o Figma também oferece uma API que ajuda a melhorar o processo de DesignOps.

Dojo
Experimente

Alguns anos atrás, o Docker — e contêineres em geral — mudaram radicalmente a forma como pensamos sobre empacotar, implantar e executar nossas aplicações. Mas, apesar dessa melhoria na produção, as pessoas desenvolvedores ainda gastam muito tempo configurando ambientes de desenvolvimento e frequentemente enfrentam problemas do tipo “funciona na minha máquina”. Dojo visa corrigir isso criando ambientes de desenvolvimento padrão, com versões, e implantados como imagens do Docker. Vários de nossos times usam o Dojo para otimizar o desenvolvimento, teste e construção de código, do desenvolvimento local aos pipelines de produção.

Adote
50. Cypress
51. Figma

Experimente
52. Dojo
53. DVC
54. Ferramentas de rastreamento de experimentos para aprendizado de máquina
55. Goss
56. Jaeger
57. k9s
58. kind
59. mkcert
60. MURAL
61. Open Policy Agent (OPA)
62. Optimal Workshop
63. Phrase
64. ScoutSuite
65. Ferramentas de teste de regressão visual
66. Visual Studio Live Share

Avalie
67. Apache Superset
68. AsyncAPI
69. ConfigCat
70. Gitpod
71. Gloo
72. Lens
73. Manifold
74. Sizzy
75. Snowpack
76. tfsec

Evite

© ThoughtWorks, Inc. All Rights Reserved.

Ferramentas

Jaeger é um sistema de rastreamento distribuído de código aberto. Nós o usamos com sucesso com Istio e Envoy no Kubernetes, e gostamos de sua UI.

(Jaeger)

O k9s fornece uma interface interativa para basicamente tudo o que o kubectl pode fazer. E, para inicializar, não é um aplicativo web, mas é executado dentro de uma janela do terminal.

(k9s)

DVC

Experimente

Em 2018, mencionamos DVC em conjunto com dados versionados para análises reproduzíveis. Desde então, ele se tornou a ferramenta preferida para gerenciar experimentos em projetos de aprendizado de máquina (ML). Baseado em Git, o DVC é um ambiente familiar para pessoas desenvolvedoras de software levarem suas práticas de engenharia à prática de ML. Como ele libera o código que processa os dados junto com os próprios dados e rastreia os estágios em um pipeline, ajuda a ordenar as atividades de modelagem sem interromper o fluxo de analistas.

Ferramentas de rastreamento de experimentos para aprendizado de máquina

Experimente

O trabalho diário de aprendizado de máquina geralmente se resume a uma série de experimentos na seleção de uma abordagem de modelagem e topologia de rede, treinamento de dados e otimização ou aprimoramento do modelo. Cientistas de dados devem usar a experiência e a intuição para criar hipóteses de mudanças e depois medir o impacto dessas mudanças no desempenho geral do modelo. À medida que essa prática amadureceu, nossos times descobriram uma demanda crescente de ferramentas de rastreamento de experimentos para aprendizado de máquina. Essas ferramentas ajudam as pessoas investigadoras a acompanhar os experimentos e trabalhar com eles de forma metódica. Embora nenhuma vencedora incontestável tenha surgido, ferramentas como MLflow e plataformas como Comet ou Neptune introduziram rigor e repetibilidade em todo o fluxo de trabalho de aprendizado de máquina.

Goss

Experimente

Mencionamos Goss, uma ferramenta para testes de provisionamento, em Radares anteriores, por exemplo, ao descrever a técnica de TDD em contêineres. Embora Goss nem sempre seja uma alternativa ao Serverspec, simplesmente por não oferecer a mesma quantidade de recursos, vale considerá-lo quando seus recursos atenderem às suas necessidades, principalmente por se tratar de um binário pequeno e independente (em vez de exigir um ambiente Ruby). Um antipadrão comum com o uso de ferramentas como Goss é a contabilidade de entrada dupla, na qual cada alteração nos arquivos de infraestrutura como código requer uma alteração correspondente nas asserções de teste. Esses testes exigem muita manutenção e, devido à estreita correspondência entre código e teste, as falhas ocorrem principalmente quando uma pessoa engenheira atualiza um lado e esquece o outro. E esses testes raramente detectam problemas genuínos.

Jaeger

Experimente

Jaeger é um sistema de rastreamento distribuído de código aberto. Semelhante ao Zipkin, foi inspirado pelo artigo do Google Dapper e está em conformidade com OpenTelemetry. Usamos Jaeger com sucesso com o Istio e o Envoy no Kubernetes e gostamos de sua UI. Jaeger expõe métricas de rastreamento no formato Prometheus, para que possam ser disponibilizadas para outras ferramentas. No entanto, uma nova geração de ferramentas como Honeycomb integra traces e métricas em um único fluxo de observabilidade para uma análise agregada mais simples. Jaeger ingressou na CNCF em 2017 e foi recentemente elevado ao mais alto nível de maturidade da CNCF, indicando sua ampla implantação em sistemas de produção.

k9s

Experimente

Continuamos apoiando fervorosamente a infraestrutura como código, e continuamos a acreditar que uma solução robusta de monitoramento é um pré-requisito para operação de aplicativos distribuídos. Às vezes, uma ferramenta interativa, como o console web da AWS, pode ser uma adição útil. Ele nos permite explorar todos os tipos de recursos de maneira ad-hoc, sem precisar lembrar todos os comandos obscuros. Entretanto, usar uma ferramenta interativa para fazer modificações manuais em tempo real ainda é uma prática questionável. Para Kubernetes, agora temos k9s, que fornece uma interface interativa para basicamente tudo o que o kubectl pode fazer. E, para inicializar, não é um aplicativo web, mas roda dentro de uma janela de terminal, evocando boas lembranças do Midnight Commander para algumas de nós.

© ThoughtWorks, Inc. Todos os direitos reservados.

kind
Experimente

kind é uma ferramenta para executar clusters locais do Kubernetes usando nós de contêiner do Docker. Com integração ao kubetest, kind facilita o teste de ponta-a-ponta no Kubernetes. Usamos o kind para criar clusters efêmeros do Kubernetes para testar recursos como Operadores e Definições de Recursos Personalizadas (CRDs) em nossos pipelines de CI.

mkcert
Experimente

mkcert é uma ferramenta conveniente para criar certificados de desenvolvimento confiáveis localmente. O uso de certificados de autoridades de certificação reais (CAs) para desenvolvimento local pode ser desafiador, se não impossível, para hosts como example.test, localhost ou 127.0.0.1. Em tais situações, certificados autoassinados podem ser sua única opção. O mkcert permite gerar certificados autoassinados e instala CA local no armazenamento raiz do sistema. Para qualquer coisa que não seja desenvolvimento e testes locais, é altamente recomendável usar certificados de autoridades de certificação reais para evitar problemas de confiança.

MURAL
Experimente

O MURAL se descreve como um "workspace digital para colaboração visual" e permite que os times interajam em um espaço de trabalho compartilhado baseado na metáfora do quadro branco com post-its. Suas funcionalidades incluem votação, comentários, notas e modo "siga o cursor de quem está apresentando". Gostamos particularmente da funcionalidade de template, que permite que uma pessoa facilitadora crie e reutilize sessões guiadas com um time. Todos os principais conjuntos de ferramentas de colaboração possuem uma ferramenta com essa finalidade (por exemplo, Google Jamboard e Microsoft Whiteboard) e vale a pena explorá-las, mas consideramos o MURAL simples, eficiente e flexível.

Open Policy Agent (OPA)
Experimente

Open Policy Agent (OPA) rapidamente se tornou um componente favorável de muitas soluções nativas de nuvem distribuídas que criamos para nossas clientes. A OPA fornece uma estrutura uniforme e linguagem para declarar, aplicar e controlar políticas para vários componentes de uma solução nativa de nuvem. É um ótimo exemplo de ferramenta que implementa política de segurança como código. Tivemos uma experiência tranquila usando OPA em vários cenários, incluindo a implantação de recursos nos clusters do K8s, impondo o controle de acesso entre serviços em uma malha de serviço e controles de segurança refinados como código para acessar os recursos da aplicação. Uma oferta comercial recente, o Serviço de Autorização Declarativa (DAS) da Styra facilita a adoção da OPA para empresas, adicionando uma ferramenta de gerenciamento ou plano de controle à OPA para K8s com uma biblioteca pré-construída de políticas, análise de impacto das políticas e recursos de log. Esperamos a maturidade e a extensão da OPA além dos serviços operacionais para (grandes) soluções centradas em dados.

Optimal Workshop
Experimente

A pesquisa de UX exige coleta e análise de dados para que sejam tomadas melhores decisões sobre os produtos que precisamos construir. Nossos times consideram o Optimal Workshop útil, pois ele facilita a validação de protótipos e a configuração de testes para coleta de dados e, portanto, a tomada de decisões melhores. Recursos como primeiro clique, classificação de cartão ou mapa de calor da interação do usuário ajudam a validar protótipos e melhoram a navegação no site e a exibição de informações. É uma ferramenta ideal para times distribuídos, pois permite realizar pesquisas remotas.

Phrase
Experimente

Conforme mencionado em nossa descrição do Crowdin, agora você tem uma opção de plataformas para gerenciar a tradução de um produto para vários idiomas, em vez de enviar grandes planilhas por email. Nossos times relatam experiências positivas com Phrase, enfatizando que é fácil de usar para todos os principais grupos de usuários. As pessoas tradutoras usam uma interface de usuário conveniente, baseada em navegador. Gerentes podem adicionar novos campos e sincronizar traduções com outros times na mesma interface de usuário. Pessoas desenvolvedoras podem acessar o Phrase localmente e a partir de um pipeline de compilação. Um recurso que merece uma menção específica é a capacidade de aplicar o controle de versão às traduções por meio de tags, o que possibilita comparar o visual de diferentes traduções no produto real.

Ferramentas

OPA fornece um framework e uma linguagem uniformes para declarar, impor e controlar políticas para vários componentes de uma solução nativa da nuvem.

(Open Policy Agent (OPA))

Nossos times consideram o Optimal Workshop útil por facilitar a validação de protótipos e a configuração de testes para coleta de dados, facilitando assim a tomada de melhores decisões.

(Optimal Workshop)

© ThoughtWorks, Inc. All Rights Reserved.

Ferramentas

AsyncAPI é uma iniciativa de código aberto para criar uma ferramenta muito necessária de padronização e desenvolvimento de API assíncrona e orientada a eventos.

(AsyncAPI)

ScoutSuite
Experimente

ScoutSuite é uma ferramenta expandida e atualizada baseada em Scout2 (destacado no Radar em 2018) que fornece avaliação de postura de segurança na AWS, Azure, GCP e outros provedores de nuvem. Ela funciona agregando automaticamente dados de configuração para um ambiente e aplicando regras para auditar o ambiente. Nós achamos isso muito útil em projetos para fazer avaliações de segurança pontuais.

Ferramentas de teste de regressão visual
Experimente

Desde que mencionamos as ferramentas de teste de regressão visual em 2014, o uso da técnica cresceu e o cenário das ferramentas evoluiu. BackstopJS continua sendo uma excelente opção, com novos recursos sendo adicionados regularmente, incluindo suporte para execução em contêineres do Docker. Loki foi destaque em nosso último Radar. Applitools, CrossBrowserTesting e Percy são soluções SaaS. Outra menção notável é Resemble.js, uma biblioteca que diferencia imagens. Embora a maioria dos times usam indiretamente como parte do BackstopJS, alguns de nossos times a usam para analisar e comparar imagens diretamente de páginas da web. Em geral, nossa experiência mostra que as ferramentas de regressão visual são menos úteis nos estágios iniciais, quando a interface passa por mudanças significativas, mas certamente provam seu valor à medida que o produto amadurece e a interface se estabiliza.

Visual Studio Live Share
Experimente

Visual Studio Live Share é um conjunto de extensões para Visual Studio Code e Visual Studio. Em um momento em que os times estão buscando boas opções para colaboração remota, queremos chamar atenção para essa excelente ferramenta. O Live Share fornece uma boa experiência de pareamento remoto de baixa latência e requer significativamente menos largura de banda do que a abordagem de força bruta ao compartilhar toda a área de trabalho. É importante ressaltar que as pessoas desenvolvedoras podem trabalhar com suas próprias configurações, extensões e mapeamentos de teclas preferidos durante as sessões de pareamento. Além da colaboração em tempo real para edição e depuração de código, o Live Share permite chamadas de voz e compartilhamento de terminais e servidores.

Apache Superset
Avalie

Apache Superset é uma excelente ferramenta de business intelligence (BI) para exploração e visualização de dados e para trabalhar com grandes configurações de lagos de dados e armazenamento de dados. Funciona, por exemplo, com Presto, Amazon Athena e Am azon Redshift e se integra bem à autenticação corporativa. Além disso, você não precisa ser especialista em engenharia de dados para usar, já que a ferramenta é feita para beneficiar todas as pessoas que trabalham com exploração de dados no dia-a-dia. Vale ressaltar que o Apache Superset está atualmente em período de incubação na Apache Software Foundation (ASF), o que significa que ainda não foi inteiramente endossado pela ASF.

AsyncAPI
Avalie

Os padrões abertos são um dos pilares fundamentais da construção de sistemas distribuídos. Por exemplo, a especificação OpenAPI (anteriormente Swagger), como padrão do setor para definir APIs RESTful, foi fundamental para o sucesso de arquiteturas distribuídas, como microsserviços. Ela permitiu uma proliferação de ferramentas para oferecer suporte à criação, teste e monitoramento de APIs RESTful. No entanto, essas padronizações têm estado ausentes em sistemas distribuídos para APIs orientadas a eventos.

AsyncAPI é uma iniciativa de código aberto para criar uma ferramenta muito necessária de padronização e desenvolvimento de API assíncrona e orientada a eventos. A especificação AsyncAPI, inspirada na especificação OpenAPI, descreve e documenta APIs controladas por eventos em um formato legível por máquinas. É independente de protocolo, por isso pode ser usada para APIs que funcionam em muitos protocolos, incluindo MQTT, WebSockets e Kafka. Esperamos ansiosamente ver as melhorias contínuas do AsyncAPI e uma maior maturidade de seu ecossistema de ferramentas.

© ThoughtWorks, Inc. Todos os direitos reservados.

ConfigCat
Avalie

Se você está procurando um serviço para suportar feature toggles dinâmicas (vale lembrar que feature toggles simples também funcionam bem), confira o ConfigCat. Nós o descreveríamos como "similar ao LaunchDarkly, mas mais barato e um pouco menos sofisticado", e consideramos que ele faz a maior parte do que precisamos. O ConfigCat suporta feature toggles simples, segmentação de usuários e teste A/B, e possui um nível gratuito generoso para casos de uso de baixo volume ou para quem está começando.

Gitpod
Avalie

Você pode criar a maioria dos softwares seguindo um processo simples de duas etapas: fazer check-out de um repositório e executar um único script de compilação. O processo de configuração de um ambiente de código completo ainda pode ser complicado. Gitpod aborda essa questão fornecendo ambientes prontos para código, baseados em nuvem, para repositórios GitHub ou GitLab. Ele oferece um IDE baseado em Visual Studio Code que roda dentro do navegador web. Por padrão, esses ambientes são lançados no Google Cloud Platform, embora você também possa implantar soluções no local. Vemos um apelo para uso imediato, especialmente para software de código aberto, onde essa abordagem pode reduzir o nível de exigência para quem colabora de maneira casual. No entanto, ainda resta ver o quanto essa abordagem será viável em ambientes corporativos.

Gloo
Avalie

Com a crescente adoção de Kubernetes e malha de serviços, os gateways de API estão enfrentando uma crise existencial nos sistemas distribuídos nativos de nuvem. Afinal, muitos de seus recursos (como controle de tráfego, segurança, roteamento e observabilidade) agora são fornecidos pelo controlador de entrada do cluster e pelo gateway de malha. Gloo é um gateway de API leve que abraça essa alteração. Ele usa Envoy como sua tecnologia de gateway, além de fornecer um valor agregado, como uma visão coesa das APIs para usuários e aplicações externas. Ele também fornece uma interface administrativa para controlar gateways Envoy e executa e integra-se a várias implementações de malha de serviço, como Linkerd, Istio e AWS App Mesh. Embora sua implementação de código aberto forneça os recursos básicos esperados de um gateway de API, sua edição corporativa possui um conjunto mais maduro de controles de segurança, como gerenciamento de chave de API ou integração com OPA. Gloo é um gateway de API leve e promissor que funciona bem com o ecossistema de tecnologia e arquitetura nativas da nuvem, evitando a armadilha do gateway de API ao permitir que a lógica de negócios cole APIs para o usuário final.

Lens
Avalie

Um dos pontos fortes do Kubernetes é sua flexibilidade e variedade de possibilidades de configuração, juntamente com os mecanismos de configuração programáveis controlados por API e a visibilidade da linha de comando e controle usando arquivos de manifesto. No entanto, essa força também pode ser uma fraqueza: quando as implantações são complexas, ou ao gerenciar vários clusters, pode ser difícil obter uma imagem nítida do status geral por meio de argumentos e manifestos da linha de comando. Lens tenta resolver esse problema com um ambiente integrado para visualizar o estado atual do cluster e suas cargas de trabalho, visualizar métricas do cluster e alterar configurações através de um editor de texto incorporado. Em vez de uma interface simples de apontar e clicar, Lens reúne as ferramentas que um administrador executaria pela linha de comando em uma única interface navegável. Essa ferramenta é uma das várias abordagens que estão tentando domar a complexidade do gerenciamento do Kubernetes. Ainda temos não temos uma vencedor indiscutível neste espaço, mas Lens encontra um equilíbrio interessante entre uma UI gráfica e ferramentas apenas de linha de comando.

Manifold
Avalie

Manifold é um depurador visual independente de modelo para aprendizado de máquina (ML). Pessoas desenvolvedores de modelos gastam uma quantidade significativa de tempo iterando e aprimorando modelos existentes, em vez de criar novos. Ao mudar o foco do espaço do modelo para o espaço dos dados, Manifold complementa as métricas de desempenho existentes com características visuais do conjunto de dados que influencia o desempenho do modelo. Acreditamos que Manifold será uma ferramenta útil para avaliar no ecossistema de ML.

Ferramentas

ConfigCat suporta feature toggles simples, segmentação de usuários e teste A/B, e possui um nível gratuito generoso para casos de uso de baixo volume ou para quem está começando.

(ConfigCat)

Lens é uma das várias abordagens diferentes para tentar domar a complexidade do gerenciamento do Kubernetes.

(Lens)

© ThoughtWorks, Inc. All Rights Reserved.

Ferramentas

tfsec é uma ferramenta de análise estática que ajuda a verificar templates do Terraform e encontrar possíveis problemas de segurança.

(tfsec)

Sizzy
Avalie

Criar aplicações web que parecem exatamente como era esperado em um grande número de dispositivos e tamanhos de tela pode ser complicado. Sizzy é uma solução SaaS que mostra muitas viewports em uma única janela do navegador. A aplicação é renderizada em todas as viewports simultaneamente e as interações com a aplicação também são sincronizadas nas viewports. Em nossa experiência, a interação com uma aplicação dessa maneira pode facilitar a identificação de possíveis problemas cedo, antes de uma ferramenta de testes de regressão visual sinalizar o problema no pipeline de compilação. Devemos mencionar, no entanto, que algumas de nossas pessoas desenvolvedores que experimentaram Sizzy por um tempo, de forma geral, preferem trabalhar com as ferramentas fornecidas pelo Chrome.

Snowpack
Avalie

Snowpack é um novato interessante no campo das ferramentas de construção em JavaScript. O principal aprimoramento em relação a outras soluções é que o Snowpack possibilita a criação de aplicativos com estruturas modernas como React.js, Vue.js e Angular sem a necessidade de um empacotador. Cortar a etapa de agrupamento melhora drasticamente o ciclo de feedback durante o desenvolvimento, porque as alterações ficam disponíveis no navegador quase imediatamente. Para que essa mágica funcione, o Snowpack transforma as dependências em `node_modules` em arquivos JavaScript únicos em um novo diretório `web_modules`, de onde eles podem ser importados como módulos ECMAScript (ESM). Para o IE11 e outros navegadores que não suportam ESM, é disponibilizada uma solução alternativa. Infelizmente, como hoje em dia nenhum navegador pode importar CSS do JavaScript, o uso de módulos CSS não é direto.

tfsec
Avalie

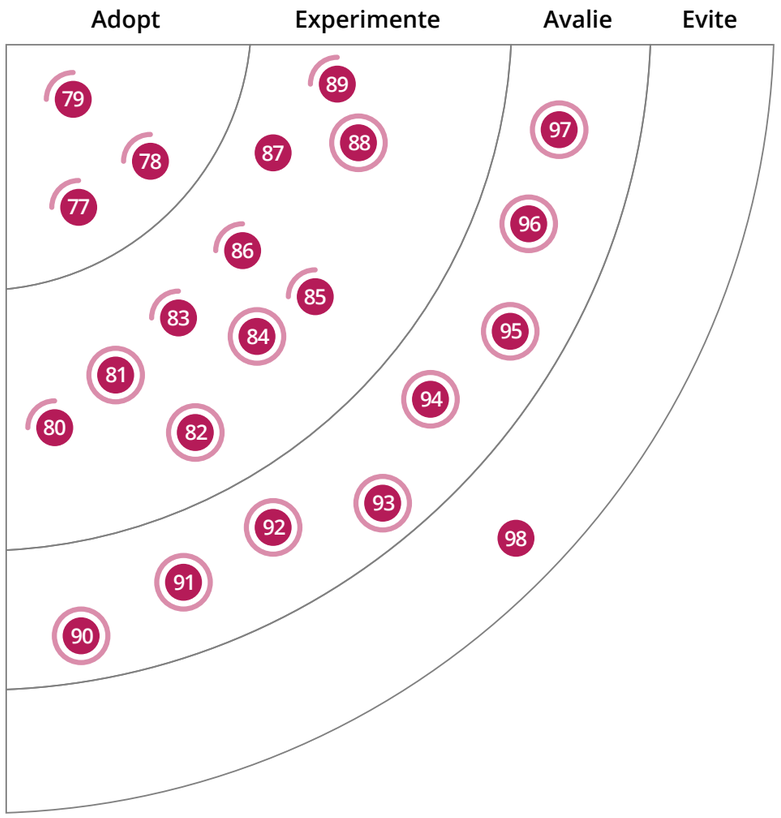
A segurança é uma preocupação geral e capturar riscos cedo é sempre melhor do que enfrentar problemas mais tarde. No espaço de infraestrutura como código, no qual Terraform é uma escolha óbvia para gerenciar ambientes em nuvem, agora também temos tfsec, uma ferramenta de análise estática que ajuda a verificar templates do Terraform e encontrar possíveis problemas de segurança. Ele vem com regras predefinidas para diferentes provedores de nuvem, incluindo AWS e Azure. Sempre gostamos de ferramentas que ajudam a mitigar os riscos à segurança, e tfsec não apenas se destaca na identificação de riscos à segurança, como também é fácil de instalar e usar.

© ThoughtWorks, Inc. Todos os direitos reservados.

TECHNOLOGY RADAR Vol. 22

Linguagens e frameworks

Linguagens e frameworks



Adote
77. React Hooks
78. Biblioteca de Testes para React
79. Vue.js

Experimente
80. CSS em JS
81. Exposed
82. GraphQL Inspector
83. Karate
84. Koin
85. NestJS
86. PyTorch
87. Rust
88. Sarama
89. SwiftUI

Avalie
90. Clinic.js Bubbleprof
91. Deequ
92. ERNIE
93. MediaPipe
94. Tailwind CSS
95. Tamer
96. Wire
97. XState

Evite
98. Enzyme

React Hooks
Adote

O React Hooks introduziu uma nova abordagem para gerenciar a lógica stateful. Como os componentes React sempre estiveram mais próximos das funções do que as classes, o Hooks adotou isso e trouxe estado para as funções, em vez de assumir as funções como métodos para o estado com classes. Com base em nossa experiência, o Hooks aprimora a reutilização da funcionalidade entre componentes e a legibilidade do código. Dadas as melhorias de testabilidade de Hooks, o uso de React Test Renderer e React Testing Library e seu crescente suporte na comunidade, os consideramos nossa abordagem de escolha.

Biblioteca de Testes para React
Adote

O mundo do JavaScript se move muito rápido e, à medida que adquirimos mais experiência usando um framework, nossas recomendações mudam. A Biblioteca de Testes para React é um bom exemplo de framework que, com um uso mais profundo, ofuscou as alternativas para se tornar o padrão razoável para testar frontends baseados em React. Nossos times gostam do fato de que os testes escritos com esse framework são menos frágeis do que os de frameworks alternativos, como Enzyme, já que ele incentiva você a testar os relacionamentos dos componentes individualmente, em vez de testar todos os detalhes da implementação. Essa mentalidade é trazida pela Biblioteca de Testes da qual a Biblioteca de Testes para React faz parte, e que também fornece uma família inteira de bibliotecas para Angular e Vue.js, por exemplo.

Vue.js
Adote

Vue.js se tornou um dos frameworks JavaScript para frontend aplicados com sucesso, amados e confiáveis em nossa comunidade. Embora existam outras alternativas bem adotadas, como React.js, a simplicidade do Vue.js no design de API, sua clara segregação de diretivas e componentes (um arquivo por idioma do componente) e seu gerenciamento de estado mais simples o tornaram uma opção atraente entre outras.

CSS em JS
Experimente

Desde que mencionamos CSS em JS como uma técnica emergente em 2017, ela

© ThoughtWorks, Inc. All Rights Reserved.

Linguagens e frameworks

Koin é um framework Kotlin que lida com um dos problemas rotineiros no desenvolvimento de software: a injeção de dependência.

(Koin)

NestJS é um framework TypeScript-first que torna o desenvolvimento de aplicações NodeJS mais seguro e menos propenso a erros.

(NestJS)

se tornou muito mais popular, e é uma tendência que também vemos em nosso trabalho. Com uma sólida experiência em produção, podemos agora recomendar CSS em JS como uma técnica a ser testada. Um bom ponto de partida é o framework de componentes com estilo, mencionado em nosso último Radar. Além de todos os aspectos positivos, porém, geralmente há uma desvantagem ao usar CSS em JS: o cálculo de estilos em tempo de execução pode causar um atraso perceptível para usuários finais. Com Linaria, agora estamos vendo uma nova classe de frameworks criados com esse problema em mente. Linaria emprega várias técnicas para mudar a maior parte da sobrecarga de desempenho para ganhar tempo. Infelizmente, isso vem com seu próprio conjunto de poréns, principalmente a falta de suporte a estilo dinâmico no IE11.

Exposed
Experimente

Com o uso prolongado de Kotlin, nossos times de desenvolvimento ganharam experiência com mais frameworks projetados especificamente para Kotlin, em vez de usar estruturas Java com Kotlin. Embora já exista há algum tempo, Exposed chamou nossa atenção como um mapeador leve de objeto relacional (ORM). Exposed possui dois tipos de acesso ao banco de dados: um DSL interno tipicamente seguro que envolve o SQL e uma implementação do padrão do objeto de acesso a dados (DAO). Ele suporta os recursos esperados de um ORM maduro, como manipulação de referências muitas-para-muitas, carregamento rápido e suporte para junções entre entidades. Também gostamos do fato de que a implementação funcione sem proxies e não dependa de reflexão, o que certamente é benéfico para o desempenho.

GraphQL Inspector
Experimente

O GraphQL Inspector permite comparar alterações entre dois esquemas de GraphQL. Nós advertimos contra o uso de GraphQL no passado, e estamos felizes em ver algumas melhorias em ferramentas de GraphQL desde então. A maioria de nossos times continua usando GraphQL para agregação de recursos do lado do servidor e, integrando o GraphQL Inspector em pipelines de CI, conseguimos captar possíveis alterações que gerem incompatibilidade no esquema.

Karate
Experimente

Dada a nossa experiência de que os testes são as únicas especificações de API que realmente importam, estamos sempre à procura de novas ferramentas que possam ajudar com testes. Karate é um framework de teste de API cuja funcionalidade exclusiva é que os testes são escritos com base na sintaxe do Gherkin, sem depender de uma linguagem de programação de uso geral para implementar o comportamento do teste. O Karate usa uma linguagem específica de domínio para descrever testes de API baseados em HTTP. Nossos times gostam da especificação legível obtida com essa ferramenta e recomendam manter os testes com Karate nos níveis superiores da pirâmide de testes, e não sobrecarregar seu uso fazendo declarações muito detalhadas.

Koin
Experimente

Com Kotlin sendo cada vez mais usado tanto para o desenvolvimento móvel quanto do lado do servidor, o ecossistema associado continua a evoluir. Koin é um framework Kotlin que lida com um dos problemas rotineiros no desenvolvimento de software: a injeção de dependência. Embora você possa escolher entre uma variedade de frameworks de injeção de dependência para Kotlin, nossos times passaram a preferir a simplicidade do Koin. Ele evita o uso de anotações e injeta por meio de construtores ou imitando a inicialização preguiçosa de Kotlin, para que os objetos sejam injetados somente quando necessário. Isso contrasta com o framework de injeção compilada estaticamente Dagger para Android. Nossas pessoas desenvolvedoras gostam da natureza leve desse framework e de sua testabilidade interna.

NestJS
Experimente

A crescente popularidade do Node.js e de tendências como uso excessivo de Node.js levaram à aplicação do Node.js no desenvolvimento de aplicações de negócios. Frequentemente, vemos problemas como escalabilidade e manutenção em aplicações grandes baseadas em JavaScript. NestJS é um framework TypeScript-first que torna o desenvolvimento de aplicações Node.js mais seguro e menos propenso a erros. O NestJS é opinativo, vem com os princípios do SOLID e uma arquitetura inspirada em Angular pronta para uso. Ao criar microsserviços Node.js, o NestJS é um dos frameworks que nossos times costumam usar para habilitar as pessoas desenvolvedoras a criar aplicações testáveis, escaláveis, com baixo acoplamento e de fácil manutenção.

PyTorch
Experimente

Nossos times continuam a usar e apreciar o framework de aprendizado de máquina PyTorch, e vários times

© ThoughtWorks, Inc. Todos os direitos reservados.

preferem PyTorch a TensorFlow. O PyTorch expõe o funcionamento interno de ML que o TensorFlow oculta, facilitando a depuração, além de conter construções com as quais as pessoas programadoras estão familiarizadas, como ciclos e ações. As versões recentes melhoraram o desempenho do PyTorch, e o usamos com sucesso em projetos de produção.

Rust
Experimente

Rust tem continuamente ganhado popularidade. Tivemos discussões acaloradas sobre o que é melhor, Rust ou C++/Go, sem um vencedor claro. No entanto, estamos felizes em ver que o Rust melhorou significativamente, com mais APIs integradas sendo adicionadas e estabilizadas, incluindo suporte assíncrono avançado, que foi mencionado no Radar anterior. Além disso, o Rust também inspirou o design de novas linguagens. Por exemplo, Move language no Libra usa a maneira do Rust de gerenciar a memória para gerenciar recursos, garantindo que os ativos digitais nunca possam ser copiados ou descartados implicitamente.

Sarama
Experimente

Sarama é uma biblioteca cliente Go para Apache Kafka. Se você estiver desenvolvendo suas APIs no Go, vai achar a Sarama muito fácil de configurar e gerenciar, pois não depende de nenhuma biblioteca nativa. Sarama possui dois tipos de APIs — uma API de alto nível para produzir e consumir mensagens facilmente e uma API de baixo nível para controlar bytes na transmissão.

SwiftUI
Experimente

A Apple deu um grande passo adiante com seu novo framework SwiftUI para implementação de interfaces de usuário nas plataformas macOS e iOS. Gostamos do fato de que o SwiftUI vá além da relação um tanto estranha entre o Interface Builder e o Xcode, e adote uma abordagem coerente, declarativa e centrada no código. Agora você pode visualizar seu código e a interface visual resultante lado a lado no Xcode 11, criando uma experiência de desenvolvimento muito melhor. O framework SwiftUI também se inspira no mundo do React.js, que dominou o desenvolvimento web nos últimos anos. Valores imutáveis nos modelos de exibição e um mecanismo de atualização assíncrona criam um modelo de programação reativa unificada. Isso fornece às pessoas desenvolvedoras uma alternativa totalmente nativa a frameworks reativos semelhantes, como React Native ou Flutter. O SwiftUI representa definitivamente o futuro do desenvolvimento da interface de usuário da Apple e, apesar de novo, demonstrou seus benefícios. Temos tido uma ótima experiência com ele — e sua curva de aprendizado superficial. Vale a pena notar que você deve conhecer o caso de uso do seu cliente antes de usar o SwiftUI, pois ele não suporta iOS 12 ou inferior.

Clinic.js Bubbleprof
Avalie

Com o objetivo de melhorar o desempenho no código, as ferramentas de criação de perfil são muito úteis para identificar gargalos ou atrasos de difícil identificação no código, especialmente em operações assíncronas. Clinic.js Bubbleprof representa visualmente as operações assíncronas nos processos do Node.js., desenhando um mapa de atrasos no fluxo da aplicação. Gostamos dessa ferramenta porque ajuda as pessoas desenvolvedoras a identificar e priorizar com facilidade o que pode ser melhorado no código.

Deequ
Avalie

Ainda existem algumas lacunas em ferramentas ao se aplicar boas práticas de engenharia de software na engenharia de dados. Tentando automatizar verificações de qualidade de dados entre diferentes etapas em um pipeline de dados, um de nossos times se surpreendeu ao encontrar apenas algumas ferramentas nesse espaço. O consenso foi Deequ, uma biblioteca para escrever testes que se assemelham a testes de unidade para conjuntos de dados. Deequ foi desenvolvida com base no Apache Spark e, mesmo sendo publicada pela AWS Labs, pode ser usada em ambientes diferentes da AWS.

ERNIE
Avalie

Na edição anterior do Radar, incluímos o BERT — que é um marco importante no cenário de PLN. No ano passado, a Baidu lançou o ERNIE 2.0 (*Enhanced Representation through kNowledge IntEgration*, ou representação aprimorada pela integração do conhecimento), que superou o BERT em sete tarefas GLUE de compreensão de linguagem e em todas as nove tarefas de PLN chinesas. O ERNIE, assim como o BERT, fornece modelos de linguagem pré-treinados não-supervisionados, que podem

Linguagens e frameworks

Deequ é uma biblioteca para escrever testes que se assemelham a testes de unidade para conjuntos de dados, o que pode ajudar na automatização de verificações de qualidade de dados entre diferentes etapas em um pipeline de dados.

(Deequ)

ERNIE provides unsupervised pretrained language models, which can be fine-tuned by adding output layers to create state-of-the-art models for a variety of NLP tasks.

(ERNIE)

© ThoughtWorks, Inc. All Rights Reserved.

Linguagens e frameworks

Tailwind CSS propõe uma abordagem interessante, fornecendo classes CSS de utilitário de nível inferior para criar blocos de construção sem estilos opinativos e visando fácil personalização.

(Tailwind CSS)

Wire é uma ferramenta de injeção de dependência em tempo de compilação que pode gerar código e conectar componentes

(Wire)

ser ajustados com a adição de camadas de saída para criar modelos de ponta para uma variedade de tarefas de PLN. O ERNIE difere dos métodos tradicionais de pré-treinamento, pois é uma estrutura de pré-treinamento contínuo. Em vez de treinar com um pequeno número de objetivos de pré-treinamento, ele pode introduzir constantemente uma grande variedade de tarefas de pré-treinamento para ajudar o modelo a aprender com eficiência as representações da linguagem. Estamos muito otimistas com os avanços em PLN e esperamos experimentar o ERNIE em nossos projetos.

MediaPipe
Avalie

MediaPipe é um framework para construção de pipelines de aprendizado de máquina aplicadas, multimodais (como vídeo, áudio, dados de séries temporais etc.), multiplataforma (por exemplo, Android, iOS, Web e dispositivos de borda). Ele fornece vários recursos, incluindo detecção facial, rastreamento de mãos, detecção de gestos e detecção de objetos. Embora o MediaPipe seja implantado principalmente em dispositivos móveis, ele começou a figurar em navegadores graças ao WebAssembly e à biblioteca de inferência XNNPack ML. Estamos explorando o MediaPipe em alguns casos de uso de realidade aumentada e gostamos do que vemos até agora.

Tailwind CSS
Avalie

As ferramentas e frameworks de CSS oferecem componentes predefinidos para resultados rápidos. Após algum tempo, no entanto, podem complicar a personalização. Tailwind CSS propõe uma abordagem interessante, fornecendo classes CSS de utilitário de nível inferior para criar blocos de construção sem estilos opinativos e visando fácil personalização. A amplitude dos utilitários de baixo nível permite evitar a criação de classes ou CSS por conta própria, o que leva a uma base de código mais sustentável a longo prazo. Aparentemente, CSS Tailwind oferece o equilíbrio certo entre reutilização e personalização para criar componentes visuais.

Tamer
Avalie

Se você precisar inserir dados de bancos de dados relacionais em um tópico do Kafka, considere o Tamer, que se autodefine como "um conector de fonte JDBC domesticado para Kafka". Apesar de ser um framework relativamente novo, achamos o Tamer mais eficiente que o conector Kafka JDBC, especialmente quando há grandes quantidades de dados envolvidas.

Wire
Avalie

A comunidade Golang teve sua parcela de pessoas céticas com injeção de dependência, em parte porque confundiram o padrão com frameworks específicos, e pessoas desenvolvedoras com experiência em programação de sistemas naturalmente não gostam da sobrecarga de tempo de execução causada pela reflexão. Então, veio o Wire, uma ferramenta de injeção de dependência em tempo de compilação que pode gerar código e conectar componentes. Wire não possui sobrecarga de tempo de execução adicional, e é mais fácil pensar no gráfico de dependência estática. Quer você escreva manualmente seu código ou use frameworks, recomendamos o uso de injeção de dependência para incentivar designs modulares e testáveis.

XState
Avalie

Já apresentamos várias bibliotecas de gerenciamento de estado no Radar, mas XState adota uma abordagem um pouco diferente. É um framework simples de JavaScript e TypeScript para criar máquinas de estados finitos e visualizá-las como grafos de estado. Ele se integra aos frameworks JavaScript reativos mais populares, (Vue.js, Ember.js, React.js e RxJS) e é baseado no padrão W3C para máquinas de estado finito. Outro recurso notável é a serialização de definições de máquina. Uma coisa que achamos útil ao criar máquinas de estados finitos em outros contextos (principalmente ao escrever a lógica) é a capacidade de visualizar estados e suas possíveis transições. Gostamos do fato de ser realmente fácil fazer isso com o visualizador do XState.

Enzyme
Evite

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

© ThoughtWorks, Inc. Todos os direitos reservados.

ThoughtWorks®

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

Quer se atualizar com artigos e informações relacionadas ao Radar?
Siga nossos perfis nas redes sociais ou torne-se assinante.

assine

ThoughtWorks®

thoughtworks.com/radar

#TWTechRadar