ThoughtWorks®

TECHNOLOGY RADAR VOL. 20

Una guía con opiniones sobre las tecnologías de vanguardia

thoughtworks.com/radar
#TWTechRadar

CONTRIBUYENTES

El Radar Tecnológico está preparado por la Junta Asesora de Tecnología de ThoughtWorks.

Esta edición del Radar Tecnológico de ThoughtWorks se basa en un encuentro de la Junta Asesora de Tecnología, que se reunió en Shenzhen en Marzo 2019

Traducción realizada por: Abraham Matus, Aldemar Santamaria, Alexandra Granda, Ana Telleria, Anna Maria Terradas, Benjamin Soto, Carlitos Oquendo, Daniel Santibañez, David Montaño, David Martín, David Corrales, Eduard Maura i Puig, Felipe Talavera, Felipe Ramirez, Glenn Wolfschoon, Gonzalo Martinez, Ivan Ortega, Javier Molina, Jesús Cardenal, Josefina Lenis Gabriel Villacis, José Puebla, Lorena Campos, Marcos Mercuri, Maria Fernanda Yépez, María José Lalama, Mateo Rojas, Melania Sanchez Blanco, Miguel Hernandez, Miquel Álvarez, Natalia Rivera, Pablo Porto, Rayner Pupo, Reynier Pupo, Stivali Serna y Tex Albuja

Rebecca Parsons (CTO)

Martin Fowler (Chief Scientist)

Bharani Subramaniam

Erik Dörnenburg

Evan Bottcher

Fausto de la Torre

Hao Xu

Ian Cartwright

James Lewis

Jonny LeRoy

Ketan Padegaonkar

Lakshminarasimhan Sudarshan

Marco Valtas

Mike Mason

Neal Ford

Ni Wang

Rachel Laycock

Scott Shaw

Shangqi Liu

Zhamak Dehghani

SOBRE EL RADAR

Los Thoughtworkers somos apasionados por la tecnología. La construimos, investigamos, probamos, liberamos su código fuente, escribimos sobre ella y constantemente queremos mejorarla para todas las personas. Nuestra misión es liderar la excelencia tecnológica y revolucionar las TI. En soporte de esta misión, creamos y compartimos el Radar Tecnológico de ThoughtWorks. La Junta Asesora de Tecnología de ThoughtWorks, un grupo de líderes senior en la tecnología dentro de ThoughtWorks, es quien crea el Radar. Se reunen regularmente para discutir la estrategia tecnológica global de ThoughtWorks y las tendencias tecnológicas que impactan significantemente nuestra industria.

El Radar captura los resultados de las discusiones de la Junta Asesora de Tecnología en un formato que provee valor a un amplio rango de personas interesadas, desde gente desarrolladora hasta CTOs. El contenido pretende ser un resumen consiso.

Los alentamos a que exploren estas tecnologías para más detalle. El Radar es gráfico por naturaleza, agrupando items en técnicas, herramientas, plataformas y lenguajes & frameworks. Cuando los items del Radar pueden aparecer en múltiples cuadrantes, elegimos el que parezca más apropiado. Después agrupamos estos items en cuatro anillos para reflejar nuestra opinión actual sobre ellos.

Para mayor información sobre el Radar, navega en thoughtworks.com/es/radar/faq

UN VISTAZO AL RADAR

1 ADOPTAR

Estamos convencidos de que la industria debería adoptar estos ítems. Nosotros los utilizamos cuando es apropiado para nuestros proyectos.

2 PROBAR

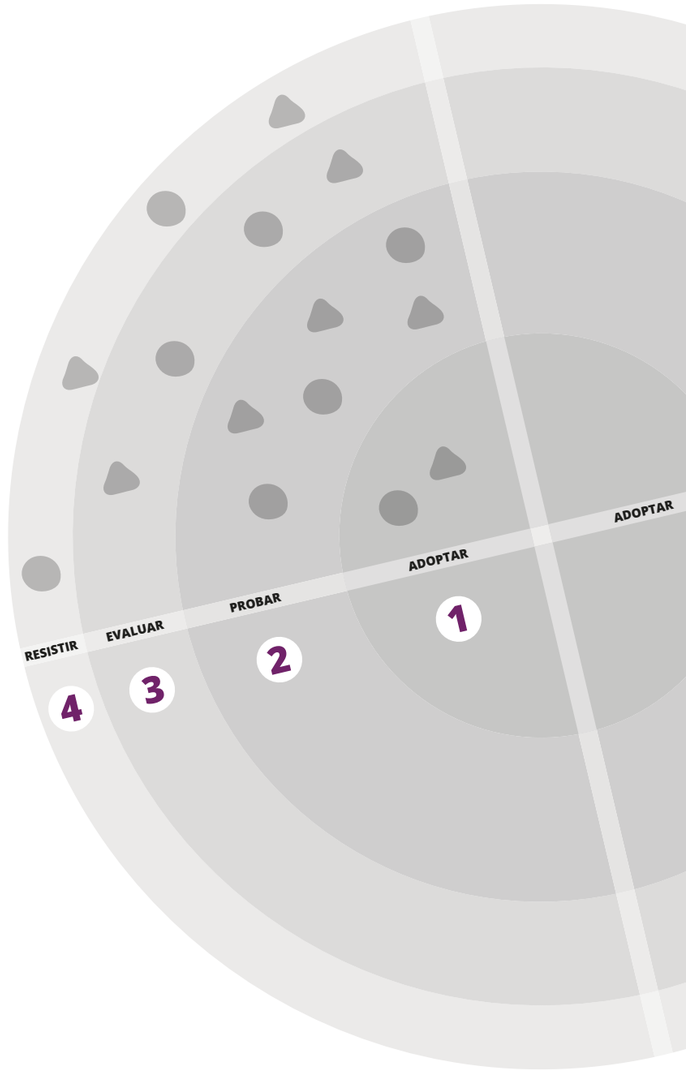
Vale la pena probarlos. Es importante entender cómo desarrollar estas capacidades. Las empresas deberían probar esta tecnología en proyectos en que se puede manejar el riesgo.

3 EVALUAR

Vale la pena explorar, con la comprensión de cómo podría afectar a su empresa.

4 RESISTIR

Proceder con precaución



▲ NUEVO O MODIFICADO
● NINGÚN CAMBIO

Los items que son nuevos o han tenido cambios significativos desde el último Radar son representados por triángulos, mientras que los items que no han cambiado son representados por círculos.

! Nuestro Radar tiene una visión hacia el futuro. Para hacer espacio a nuevos items, hemos retirado los items que no han sufrido cambios recientes, lo cual no es un reflejo de su valor sino más bien del espacio limitado disponible en nuestro Radar.

QUÉ HAY DE NUEVO

Temas destacados en esta edición

El Cambiante Panorama de los Datos

Hace una década, los datos eran sinónimos de bases relacionales. Ahora los datos pueden tomar una asombrosa variedad de formas incluyendo NoSQL, series de tiempo, almacenes SQL como CockRoachDB y Spanner que ofrecen consistencia global así como transmisiones de eventos que habilitan consultas de archivos de logs agregados. Esto es dirigido por los deseos del negocio de obtener respuestas en tiempo real sobre orígenes de datos cada vez más grandes, variados y rápidos. Para los desarrolladores, el entendimiento del balance inherente entre cada sabor del uso de los datos puede plantear desafíos. Los arquitectos y desarrolladores deberían estar atentos a nuevas capacidades ofrecidas por las nuevas herramientas y paradigmas familiarizándose con ellas y manteniéndose diligentes para no hacer un mal uso de las mismas. Deberíamos aceptar el hecho de que estamos en medio de un gran cambio en el panorama de datos para así mantenernos en búsqueda de estrategias y herramientas.

El Ecosistema Terraform

Los desarrolladores aman las capas de abstracción por obvias razones. El encapsulamiento de la complejidad en una abstracción en la que ellos puedan concentrarse en preocupaciones de más alto nivel. Hemos visto esta evolución a través de muchas ediciones del Radar en la medida en que los equipos tratan con las intersecciones entre nubes y contenedores. En primer lugar, los esfuerzos enfocados en docker y su ecosistema. Luego, el enfoque cambió el stack a Kubernetes. Ahora la principal actividad que vemos está en la infraestructura como código en general y el ecosistema Terraform en particular. A pesar de que hemos recomendado herramientas más allá de Terraform, su adopción en la comunidad de proveedores ha sido impresionante. Lo más destacado en este Radar incluye Terratest para probar infraestructura como código y GoCD's new Provider, el cual permite configurar GoCD usando Terraform.

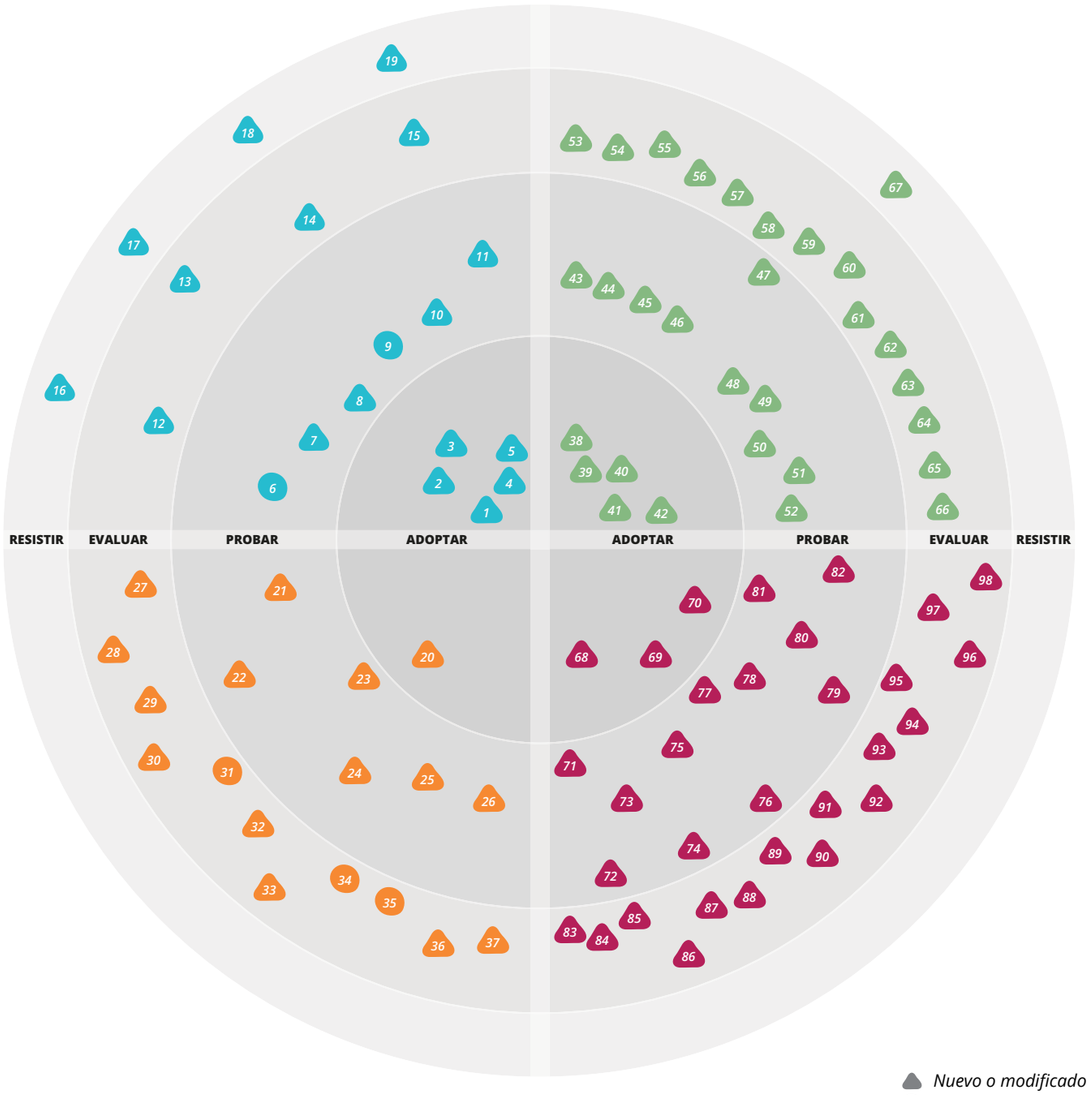
Krecimiento de Kotlin

Kotlin, el lenguaje de código abierto, sigue teniendo fuerte presencia en nuestro Radar mientras se expande sobre la fortaleza de Android. Fue creado internamente en JetBrains, porque no les gustaba las opciones disponibles en el paisaje lingüístico existente, y seguidamente fue lanzado bajo una licencia de código libre, Kotlin parece hacer ruido entre un gran abanico de desarrolladores. Continúa apareciendo en plataformas y herramientas como un lenguaje de propósito general y específico, y de forma incremental en nuestro Radar y en nuestros equipos de trabajo (Ej., Ktor, MockK, Detekt, HTTP4K). Es alentador ver que, un diseño pragmático, una herramienta de última generación, y un ecosistema en vías de expansión pueda hacer triunfar a un lenguaje emergente.

Rompiendo los Límites de la Encapsulación

Con la llegada de "todo como código", casi todo (infraestructura, seguridad, cumplimiento y operaciones) lo que antes era difícil de cambiar se convierte automáticamente en fácil de abordar, lo que significa que los desarrolladores podrían aplicar buenas prácticas de programación; sin embargo aún vemos con bastante frecuencia, subsistemas de configuración complejos, una gran dependencia en herramientas de dirección visual, introducción de lógica en archivos de configuración, sintaxis difícil de comprender en condicionales YAML, y muchos marcos de trabajo de organización a través de una gran variedad de tecnologías. Con la llegada de la programación políglota, infraestructura como código y x-como-servicio, los equipos terminan con diversos componentes que se fusionan en un único sistema cohesionado. Por lo tanto, la lógica que debería residir dentro del sistema se decanta hacia las herramientas de dirección, los archivos de configuración, y otras instalaciones. Mientras esto es algunas veces necesario, animamos a los equipos a que consideren concienzudamente el mantener dicho código en lugares donde los desarrolladores se ciñan a mantener tests, control de versiones, integración continua, y otras buenas prácticas de programación. Evitar añadir lógica de negocio en los archivos de configuración y como consecuencia evitar herramientas que lo requieran, y preferir la orquestación al mínimo en lugar de una característica predominante en un sistema.

EL RADAR



TÉCNICAS

ADOPTAR
1. Cuatro métricas clave
2. Micro frontends
3. Formateo de código automatizado con opinión
4. Programación políglota
5. Secretos como servicio

PROBAR
6. Ingeniería del Caos
7. Análisis de seguridad en contenedores
8. Entrega continua para modelos de machine learning (CD4ML)
9. Crypto shredding
10. Análisis de la configuración de la infraestructura
11. Malla de Servicios

EVALUAR
12. Ethical OS
13. Contratos Inteligentes
14. Transferencia de aprendizaje para NLP
15. Wardley mapping

RESISTIR
16. Productionizing Jupyter Notebooks
17. Perforar la encapsulación capturando los cambios en los datos
18. Release train
19. Plantillas en YAML

PLATAFORMAS

ADOPTAR
20. Contentful

PROBAR
21. AWS Fargate
22. EVM más allá de Ethereum
23. InfluxDB
24. Istio
25. Kafka Streams
26. Nomad

EVALUAR
27. CloudEvents
28. Cloudflare Workers
29. Deno
30. Hot Chocolate
31. Knative
32. MinIO
33. Prophet
34. Quorum
35. SPIFFE
36. Tendermint
37. TimescaleDB

RESISTIR

HERRAMIENTAS

ADOPTAR
38. Cypress
39. Jupyter
40. LocalStack
41. Terraform
42. UI dev environments

PROBAR
43. AnyStatus
44. AVA
45. batect
46. Elasticsearch LTR
47. Helm
48. InSpec
49. Lottie
50. Stolon
51. TestCafe
52. Traefik

EVALUAR
53. Anka
54. Cage
55. Cilium
56. Detekt
57. Flagr
58. Gremlin
59. Honeycomb
60. Humio
61. Operadores Kubernetes
62. OpenAPM
63. Systems
64. Taurus
65. Terraform provider GoCD
66. Terratest

RESISTIR
67. Handwritten CloudFormation

LENGUAJES & FRAMEWORKS

ADOPTAR
68. Apollo
69. MockK
70. TypeScript

PROBAR
71. Apache Beam
72. Formik
73. HiveRunner
74. joi
75. Ktor
76. Laconia
77. Puppeteer
78. Reactor
79. Resilience4j
80. Room
81. Rust
82. WebFlux

EVALUAR
83. Aeron
84. Arrow
85. Chaos Toolkit
86. Dask
87. Embark
88. fastai
89. http4k
90. Immer
91. Karate
92. Micronaut
93. Next.js
94. Pose
95. react-testing-library
96. ReasonML
97. Taiko
98. Vapor

RESISTIR

TÉCNICAS

Cuatro métricas clave
ADOPTAR

Los reportes exhaustivos State of DevOps se han enfocado en los análisis guiados por datos y estadísticas de organizaciones de alto rendimiento. El resultado de esta investigación de varios años, publicado en Accelerate, demuestra un vínculo directo entre el rendimiento organizacional y el rendimiento de entrega de software. Los investigadores del estudio han determinado que solo cuatro métricas clave diferencian un rendimiento bajo de uno medio o alto: tiempo de ejecución, frecuencia de despliegue, tiempo promedio de restauración (MTTR) y porcentaje de fallos por cambios. En efecto, hemos encontrado que estas 4 métricas clave son una herramienta simple pero poderosa para ayudar a líderes y equipos a centrarse en medir y mejorar lo que importa. Un buen comienzo es instrumentar las pipelines de compilación para capturar las cuatro métricas clave y hacer visible el flujo de la entrega de software. GoCD pipelines, por ejemplo, provee la habilidad de medir estas cuatro métricas clave como ciudadanos de primera clase en GoCD analytics.

Micro frontends
ADOPTAR

Hemos observado beneficios significativos al introducir microservicios, que han permitido a los equipos escalar la entrega de servicios independientemente desplegados y mantenidos. Desafortunadamente, también hemos visto a muchos equipos crear un frontend monolítico — una aplicación grande y complicada en el navegador montada sobre servicios backend — que neutraliza en gran medida los beneficios de tener microservicios.
Desde que describimos por primera vez micro frontends como técnica para atacar este problema, hemos tenido muchas experiencias positivas con este enfoque y hemos encontrado una serie de patrones para usar micro frontends incluso a medida que más y más código se mueve del servidor al navegador. Sin embargo, los componentes web (web components) se han mostrado esquivos en este campo por el momento.

Formateo de código automatizado con opinión
ADOPTAR

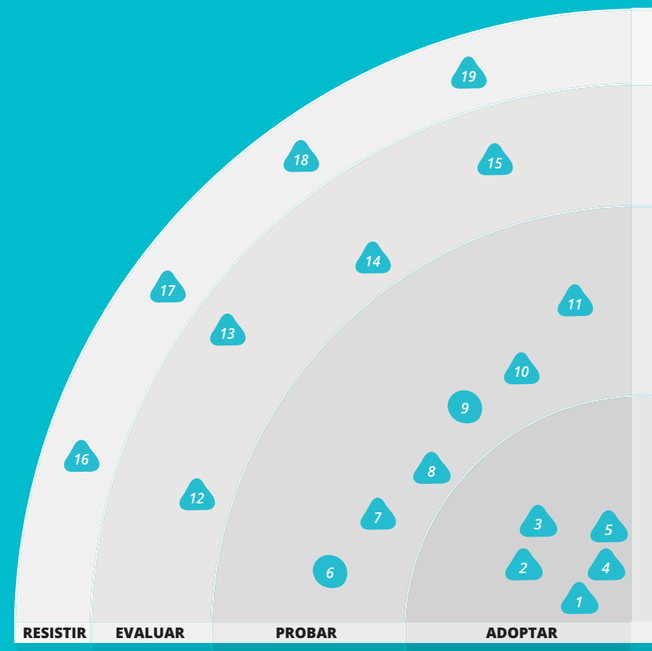
Desde que tenemos uso de razón, qué estilo de formateo de código se usa ha sido un asunto de preferencia personal, política de compañía y debate acalorado. Finalmente, la industria parece haberse cansado de la interminable discusión y los equipos están liberando una cantidad sorprendentemente grande de tiempo absteniéndose de estas discusiones y adoptando herramientas de formateo de código sesgado y automatizado. Aunque no estés 100% de acuerdo con las opiniones de las distintas herramientas, el beneficio de centrarse en lo que el código hace en vez de cómo queda es algo que mucho equipos deberían apoyar.

ADOPTAR
1. Cuatro métricas clave
2. Micro frontends
3. Formateo de código automatizado con opinión
4. Programación políglota
5. Secretos como servicio

PROBAR
6. Ingeniería del Caos
7. Análisis de seguridad en contenedores
8. Entrega continua para modelos de machine learning (CD4ML)
9. Crypto shredding
10. Análisis de la configuración de la infraestructura
11. Malla de Servicios

EVALUAR
12. Ethical OS
13. Contratos Inteligentes
14. Transferencia de aprendizaje para NLP
15. Wardley mapping

RESISTIR
16. Productionizing Jupyter Notebooks
17. Perforar la encapsulación capturando los cambios en los datos
18. Release train
19. Plantillas en YAML



TÉCNICAS

Elegir el lenguaje correcto para el problema a solucionar puede aumentar significativamente la productividad, y promover lenguajes que apoyen ecosistemas es importante para que los sistemas funcionen rápidamente y para que los desarrolladores tengan herramientas para abordar problemas adecuadamente.

(Programación políglota)

Prettier ha estado obteniendo nuestro voto para JavaScript, aunque herramientas similares, como Black para Python, están disponibles para muchos otros lenguajes y se construyen incrementalmente como vemos con Golang y Elixir. La clave aquí es no gastar horas discutiendo cuáles reglas cumplir, sino elegir una herramienta con opinión, mínimamente configurable y automatizada - idealmente como un pre-commit hook.

Programación políglota

ADOPTAR

Pusimos programación políglota en "probar" en uno de nuestras primeras ediciones del Radar para sugerir que elegir el lenguaje correcto para el problema a solucionar puede aumentar significativamente la productividad, y puesto que, además, había nuevos lenguajes que valía la pena considerar. Queremos traer nuevamente esta sugerencia porque estamos viendo nuevos esfuerzos de estandarizar los stack de lenguajes por parte tanto de desarrolladores como de empresas. Si bien reconocemos que el hecho de no poner restricciones en el uso de distintos lenguajes puede crear más problemas de los que resuelve, promover unos pocos lenguajes que apoyen distintos ecosistemas o funcionalidades es importante para ambas partes; para las compañías para acelerar los procesos y entregar valor rápidamente y para los desarrolladores para tener las herramientas correctas para resolver los problemas a mano.

Secretos como servicio

ADOPTAR

Tanto los humanos como las máquinas usan secretos en la cadena de valor de la construcción y operación del software. Los pipelines de construcción necesitan secretos para comunicarse con infraestructuras seguras como son los registros de contenedores, las aplicaciones usan llaves de API como secretos para tener acceso a funcionalidades de negocio y la comunicación servicio-servicio se asegura usando certificados y llaves. Estos secretos se pueden establecer y obtener de varias maneras. Hace tiempo se le ha advertido a los desarrolladores acerca del uso de repositorios de código fuente para el almacenamiento de secretos. Hemos recomendado desacoplar el manejo de secretos del código fuente y utilizar herramientas como git-secrets y Talisman para evitar el almacenamiento de secretos en el código. Igualmente, hemos estado utilizando secretos como servicio como una técnica para almacenar y acceder a los secretos. Mediante esta técnica podemos emplear herramientas como Vault o AWS Key Management Service (KMS) para leer o escribir secretos sobre un endpoint HTTPS con niveles de acceso bien definidos. Los secretos como servicio utilizan proveedores de entidades externos como AWS IAM para identificar los actores que hacen las peticiones para acceder a los secretos; estos actores se autentican a sí mismos mediante servicios de secretos. Para que este proceso funcione es importante automatizar la creación de la identidad de los actores, servicios y aplicaciones. Las plataformas SPIFFE han mejorado la automatización de la asignación de identidades a los servicios.

Ingeniería del Caos

PROBAR

En el último año hemos visto que la ingeniería del caos ha evolucionado de ser una idea de la que se hablaba mucho a ser una idea aceptada y usada por la mayoría para mejorar y asegurar la resiliencia de los sistemas distribuidos. Dado que grandes y pequeñas organizaciones han comenzado a implementar la ingeniería del caos como un proceso operacional, estamos aprendiendo cómo aplicar estas técnicas de forma segura a gran escala. El enfoque no es definitivamente para todos, y para que sea efectivo y seguro, requiere de apoyo organizacional a gran escala. La aceptación en la industria y la experiencia disponible aumentará definitivamente con la aparición de servicios comerciales tales como Gremlin y herramientas de desarrollo como Spinnaker, que implementan algunas de las herramientas de la ingeniería del caos.

Análisis de seguridad en contenedores

PROBAR

La revolución del contenedor alrededor de Docker ha reducido enormemente la fricción de mover aplicaciones entre ambientes, alimentando una mayor adopción de la entrega e implementación continua. Especialmente esta última, ha abierto un agujero bastante grande en el control tradicional sobre lo que puede ir a producción. La técnica de análisis de seguridad en contenedores es una respuesta necesaria a este vector de amenaza. Las herramientas en el pipeline verifican automáticamente los contenedores contra las vulnerabilidades conocidas. Desde nuestra primera mención de esta técnica, el horizonte de herramientas ha madurado y la técnica ha demostrado ser útil en los esfuerzos de desarrollo con nuestros clientes.

Entrega continua para modelos de machine learning (CD4ML)

PROBAR

Los modelos de entrega continua de machine learning (CD4ML) aplica las prácticas de entrega continua para desarrollar modelos de machine learning que permitan estar siempre listos para producción. Esta técnica direcciona dos problemas principales de modelo de desarrollo tradicional de machine learning: largo tiempo de ciclo entre los modelos de entrenamiento y el despliegue a producción, lo que frecuentemente incluye conversión manual al modelo de código-listo para producción; y el uso de modelos de producción que han sido entrenados con data pasada.

Un modelo de entrega continua de machine learning tiene dos disparadores: (1) cambios a la estructura del modelo y (2) cambios al entrenamiento y prueba del conjunto de datos. Para que esto trabaje, necesitamos tanto el versionamiento del conjunto de datos como un código seguro del modelo. El camino incluye frecuentemente pasos como probar el modelo contra el conjunto de datos de prueba, aplicando conversiones automáticas del modelo (si es necesario) con herramientas como H2O y desplegar el modelo a producción para agregar valor.

Crypto shredding

PROBAR

Mantener el control apropiado sobre datos confidenciales es complicado, especialmente cuando se copian fuera de un sistema de registro principal con el fin de respaldar y recuperar. Crypto shredding es una práctica de hacer ilegibles los datos confidenciales, sobreescribiendo o eliminando deliberadamente las claves de encriptación usadas para asegurar dichos datos. Considerando los sistemas actuales, como aplicaciones de auditoría o blockchain, que no pueden o no deben, eliminar archivos históricos, esta técnica es bastante útil para la protección de la identidad y el cumplimiento de GDPR.

Análisis de la configuración de la infraestructura

PROBAR

Desde hace algún tiempo hemos recomendado que los equipos de desarrollo se hagan dueños de todo su stack, incluyendo la infraestructura. Esto significa aumentar la responsabilidad que los equipos tienen a la hora de configurar la infraestructura de forma segura, protegida y de acuerdo a las normativas. Cuando se adoptan estrategias basadas en la nube, la mayoría de las organizaciones tienden por defecto a crear configuraciones fuertemente cerradas y manejadas de forma central para reducir riesgos, pero esto causa grandes cuellos de botella que afectan la productividad. Un enfoque alternativo es permitir a los equipos gestionar sus propias configuraciones y utilizar herramientas de análisis de configuración de la infraestructura para asegurar que dichas configuraciones están seguras y protegidas. Algunas opciones incluyen escáneres open-source como prowler para AWS y kube-bench para Kubernetes. Para conseguir detección continua, puedes echar un vistazo a servicios como AWS Config Rules entre otros servicios comerciales.

Malla de Servicios

PROBAR

La Malla de Servicios es una solución para operar en un ecosistema de microservicios de forma segura, rápida y confiable. Ha sido un escalón importante para facilitar la adopción de microservicios a gran escala. Ofrece el descubrimiento, seguridad, trazabilidad, monitoreo y manejo de fallas. Provee esas capacidades interfuncionales sin la necesidad de recursos compartidos como pasarelas de API o exponer librerías en cada servicio. Una implementación común involucra procesos de un proxy inverso ligero, también conocidos como Sidecars, desplegado junto a cada proceso de servicio en un contenedor separado. Los Sidecars interceptan el tráfico entrante y saliente de cada servicio y proveen las funcionalidades previamente mencionadas. Este alcance ha aliviado a los equipos de servicios distribuidos de construir y actualizar las capacidades que las mallas ofrecen como código en sus servicios. Esto ha llevado a una adopción más fácil de la programación políglota en ecosistemas de microservicios. Nuestros equipos han usado con éxito esta solución con proyectos de código abierto como Istio y continuaremos monitoreando de cerca otras implementaciones de mallas de servicios como Linkerd.

Ethical OS

EVALUAR

Como desarrolladores en Thoughtworks, somos conscientes de la ética del trabajo que realizamos. A medida que la sociedad se vuelve más dependiente de la tecnología, es relevante que consideremos la ética como

TÉCNICAS

El desarrollo tradicional de modelos de Machine Learning tiende a tener largos tiempos de ciclo entre los modelos de entranamiento y, finalmente, implementándolos a producción, y a menudo terminamos con modelos de producción entrenados con (ahora) datos del pasado. Aplicaccar prácticas de entrega continua a Machine Learning puede abordar ambos problemas.

(Entrega continua para modelos de machine learning (CD4ML))

A medida que la sociedad se vuelve más dependiente de la tecnología, es relevante que consideremos la ética como parte de la toma de decisiones de nuestros equipos. Ethical OS es un patrón de pensamiento que incluye un conjunto de herramientas que generan discusiones sobre la ética de desarrollar software.

(Ethical OS)

TÉCNICAS

Los datos inmutables son una cosa, pero la lógica empresarial inmutable es algo completamente distinto — pensar detenidamente antes de comprometer la lógica empresarial con un contrato inteligente.

(Contratos inteligentes)

parte de la toma de decisiones de nuestros equipos. Han surgido variadas herramientas que pueden ayudarnos a pensar en las futuras implicaciones del software que desarrollamos; por ejemplo, Tarot Cards of Tech y Ethical OS, de las cuales hemos recibido reacciones positivas. Ethical OS es un patrón de pensamiento que incluye un conjunto de herramientas que generan discusiones sobre la ética de desarrollar software. Es producto de una colaboración entre dos organizaciones, Institute for the Future y Tech and Society Solutions Lab. Está basado en un conjunto práctico de zonas de riesgo, como la adicción y la economía que se aprovecha de la misma (dopamine economy), además de un número de situaciones que conllevan a conversaciones y discusiones.

Contratos Inteligentes

EVALUAR

Cuanta más experiencia ganamos usando tecnologías de contabilidad distribuida (DLTs), más nos encontramos con los límites en torno al estado actual de los contratos inteligentes. Elaborar contratos automatizados, irrefutables e irreversibles en el libro mayor suena muy bien en teoría. Los problemas surgen cuando se considera cómo utilizar las técnicas modernas de entrega de software para desarrollarlas, así como diferencias entre las implementaciones. Los datos inmutables son una cosa, pero la lógica empresarial inmutable es algo completamente distinto. Es muy importante pensar si se debe incluir lógica en un contrato inteligente. También hemos encontrado características operativas muy diferentes entre las implementaciones. Por ejemplo, aunque los contratos pueden evolucionar, diferentes plataformas soportan esta evolución en mayor o menor medida. Nuestro consejo es pensar detenidamente antes de comprometer la lógica empresarial con un contrato inteligente y sopesar los beneficios de cada plataforma antes de hacerlo.

Transferencia de aprendizaje para NLP

EVALUAR

La transferencia del conocimiento ya sido bastante efectiva en el cambio de la visión computarizada, acelerando el tiempo de entrenamiento de un módelo reutilizando estos módulos existentes. Para los aquellos que trabajamos en machine learning estamos emocionados que esas mismas técnicas puedan ser aplicadas al procesamiento de lenguaje natural (NLP) con la publicación de ULMFiT, modelos pre-entrenados y ejemplos de código. Pensamos que la transferencia de aprendizaje para NLP reducirá significativamente el esfuerzo de crear sistemas y lidiar con la clasificación de textos.

Wardley mapping

EVALUAR

Normalmente somos cautelosos a la hora de considerar técnicas esquemáticas, pero creemos que Wardley mapping es un enfoque interesante para empezar a tener conversaciones sobre la evolución del estado del software en una organización. En su forma más simple, se utiliza para visualizar las cadenas de valor que existen dentro de una organización, empezando por las necesidades de los clientes y trazando progresivamente las diferentes capacidades y sistemas utilizados para satisfacer esas necesidades junto con la evolución de esas capacidades y sistemas. El valor de esta técnica está en el proceso de colaboración para crear mapas en vez del artefacto en sí. Se recomienda tener a la gente adecuada en la habitación para producirlos, y luego tratarlos como algo vivo y evolutivo en vez de como artefactos completos.

Productionizing Jupyter Notebooks

RESISTIR

Los Jupyter Notebooks han ganado popularidad entre los científicos de datos, quienes los utilizan para análisis exploratorios, desarrollo en etapas iniciales e intercambio de conocimientos. Este alza en la popularidad ha conducido a un aumento en la tendencia de llevar Jupyter Notebooks a producción, al proporcionar las herramientas y el soporte para ejecutarlos a gran escala. Aunque no queremos desalentar a nadie a que use las herramientas que deseen, no recomendamos el uso de Jupyter Notebooks para crear códigos de producción escalables, mantenibles y de larga duración, ya que carecen de un efectivo control de versiones, manejo de errores, modularidad, y extensibilidad, entre otros. Capacidades básicas requeridas para construir código escalable y listo para producción. En su lugar, alentamos a los desarrolladores y científicos de datos a trabajar juntos para encontrar soluciones que permitan a estos últimos a construir modelos de aprendizaje automático listos para producción, utilizando prácticas de entrega continua con los marcos de programación adecuados. Advertimos contra la puesta en producción de Jupyter Notebooks, para superar las ineficiencias en las líneas de entrega continua para el aprendizaje automático, o las pruebas automatizadas inadecuadas.

Perforar la encapsulación capturando los cambios en los datos

RESISTIR

La captura de cambio de datos - Change data capture (CDC) - es una técnica muy poderosa para extraer cambios en bases de datos de un sistema y realizar acciones sobre estos datos. Una de las maneras más comunes de hacerlo es usar los logs de las transacciones de la base de datos para identificar cambios y después publicar esos cambios directamente en un bus de eventos que pueden ser consumidos por otros servicios. Este mecanismo funciona muy bien para casos de uso como romper monolitos en microservicios pero cuando se usa para integración entre microservicios, esto conlleva perforar la encapsulación y filtrar la capa de datos del servicio en el contrato del evento. Hemos hablado de eventos con ámbitos de dominio y otras técnicas que enfatizan la importancia de hacer que nuestros eventos modelen nuestro dominio apropiadamente. Vemos algunos proyectos usar el CDC para publicar eventos de cambios a nivel de fila y consumir directamente estos eventos en otros servicios. Este perforamiento de encapsulación con captura de cambio de datos puede presentar situaciones resbaladizas que resulten en integraciones frágiles y nos gustaría comunicarlo con este blip..

Release train

RESISTIR

Hemos observado organizaciones que se han movido satisfactoriamente desde pasos a producción muy poco frecuentes a una alta cadencia usando el concepto de release train. El release train es una técnica para coordinar entregas entre múltiples equipos o componentes que tienen una dependencia de ejecución. Todos los pasos a producción ocurren en un horario fijo y de confianza independientemente de si las funcionalidades esperadas están listas o no (el tren no espera — si lo pierdes tienes que esperar al siguiente). Aunque respaldamos incondicionalmente la disciplina de lanzar y mostrar software funcional de manera regular, hemos experimentado serios inconvenientes con el enfoque a medio y largo plazo, ya que refuerza el acoplamiento temporal en torno a la secuencia de los cambios y puede degradar la calidad a medida que los equipos se apresuran a terminar dentro de un tiempo limitado.

Preferimos centrarnos en los enfoques arquitectónicos y organizativos necesarios para soportar liberaciones independientes. Aunque el tren puede ser una función útil para obligar a acelerar a equipos lentos, también lo vemos como un límite para aquellos equipos que ya trabajan con cierta velocidad. En todo caso, creemos que es una técnica que debe abordarse con un cierto grado de precaución.

Plantillas en YAML

RESISTIR

A medida que las infraestructuras crecen en complejidad, también lo hacen los archivos de configuración que los definen. Herramientas como AWS CloudFormation, Kubernetes y Helm esperan archivos de configuración con sintaxis de JSON o YAML, presumiblemente en un intento de hacer más fácil la escritura y el proceso. Sin embargo, en muchos casos, cuando un mismo servicio debe ser desplegado en diferentes regiones con ligeras diferencias de configuración. Para cada caso, las herramientas ofrecen plantillas en YAML (o JSON), las cuales han ocasionado enormes cantidades de frustración profesional. El problema es que la sintaxis de JSON y YAML requieren todo tipo de concesiones para insertar características de las plantillas como condiciones y loops a los archivos. Nuestra recomendación es utilizar una API desde un lenguaje de programación en lugar o, cuando no es opcional, un sistema de plantillas desde un lenguaje de programación, ya sea un lenguaje de propósito general como Python o algo especializado tal como Jsonnet.

TÉCNICAS

Usar change data capture (CDC) para publicar eventos de cambios a nivel de fila y consumir directamente estos eventos en otros servicios puede generar integraciones frágiles

(Perforar la encapsulación capturando los cambios en los datos)

Aunque respaldamos incondicionalmente la disciplina de lanzar y mostrar software funcional de manera regular, hemos experimentado serios inconvenientes con el enfoque a medio y largo plazo, ya que refuerza el acoplamiento temporal en torno a la secuencia de los cambios y puede degradar la calidad a medida que los equipos se apresuran a completar las funciones.

(Release train)

PLATAFORMAS

Contentful
ADOPTAR

Los sistemas de gestión de contenido "headless" (CMS, por sus siglas en inglés) cada vez son un componente más habitual en las plataformas digitales. Contentful es un CMS "headless" moderno que nuestros equipos han integrado en sus ciclos de desarrollo. Nos gusta especialmente su enfoque "API-first" y su implementación de CMS como código. Da soporte a potentes primitivas en forma de código para el modelado de contenido y a "scripts" para la evolución de los modelos de contenido, que nos permiten tratarlo como otros esquemas de almacenamiento de datos y posibilitan la aplicación de prácticas de diseño de bases de datos evolutivas al desarrollo de CMS. Su solidez y una oleada de nuevas funcionalidades, que incluyen un entorno de pruebas, han impresionado a nuestros equipos aún más, convirtiendo a Contentful en nuestra elección por defecto de este espacio.

AWS Fargate
PROBAR

AWS Fargate, la opción de AWS de docker-as-a-service, está ahora disponible completamente a través de distintas regiones. Es una excelente solución para los casos en aquellos equipos que quieren correr contenedores Docker, porque las funciones AWS Lambda no son lo suficientemente poderosas, sin tener que gestionar instancias EC2 o clusters en Kubernetes. Nuestros equipos informan experiencias positivas generalmente con Fargate; sin embargo la conveniencia de manejar este servicio puede venir con un costo, en términos financieros.

EVM más allá de Ethereum
PROBAR

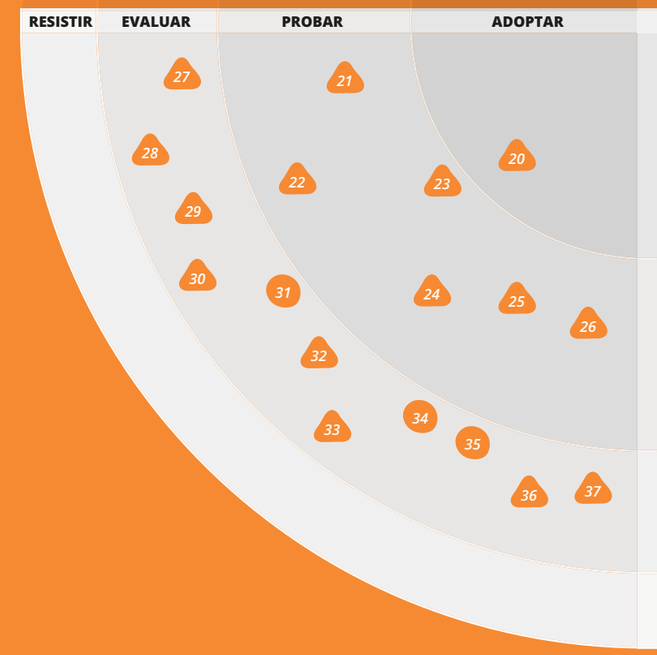
Ethereum Virtual Machine (EVM) fue originalmente diseñada por la red principal de Ethereum. Sin embargo, actualmente la mayoría de los equipo no quiere seguir reinventando blockchain desde cero; En su lugar, quieren llevar EVM más allá de Ethereum. Hemos visto mucho de los equipos de blockchain elegir entre hacer fork a Ethereum (e.g., Quorum) o implementar la especificación EVM (e.g., Burrow, Pantheon), agregando sus propios diseños. La intención no es solo reusar el diseño de Ethereum sino influenciar su ecosistema y la comunidad de las personas desarrolladoras. Para muchas personas desarrolladoras, el concepto de "contratos inteligentes" es casi equivalente a un contrato inteligente escrito en Solidity. Sin embargo, Ethereum en sí mismo tiene muchas restricciones, la tecnología alrededor del ecosistema de EVM está en auge.

ADOPTAR
20. Contentful

PROBAR
21. AWS Fargate
22. EVM más allá de Ethereum
23. InfluxDB
24. Istio
25. Kafka Streams
26. Nomad

EVALUAR
27. CloudEvents
28. Cloudflare Workers
29. Deno
30. Hot Chocolate
31. Knative
32. MinIO
33. Prophet
34. Quorum
35. SPIFFE
36. Tendermint
37. TimescaleDB

RESISTIR



PLATAFORMAS

Las bases de datos de series de tiempo han existido desde hace algún tiempo, pero se están haciendo más populares a medida que más casos de uso calzan de manera natural en el modelo de series de tiempo. InfluxDB igue siendo una buena elección en este espacio.

(InfluxDB)

Aunque los eventos son un mecanismo usual para ejecutar las FaaS, y todos los proveedores de servicios en la nube los soportan de alguna manera, las especificaciones propietarias actuales impiden su interoperatividad entre nubes. CloudEvents es un estándar creciente que intenta abordar este problema.

(CloudEvents)

InfluxDB

PROBAR

Las bases de datos de series de tiempo (TSBDs - Time Series Databases) han existido desde hace algún tiempo. Pero se están haciendo más populares a medida que más casos de uso calzan de manera natural en el modelo de series de tiempo. InfluxDB sigue siendo una buena elección de TSDB, ya que la monitorización es uno de sus casos de uso clave. TICK Stack es un ejemplo de una solución de monitorización que tiene a InfluxDB como núcleo. Influx 2.0 alpha recientemente introdujo Flux, un lenguaje de secuenciamiento de comandos (scripting) para consultar y procesar datos de series de tiempo. Flux está aún en sus primeros días y aún no hay consenso acerca de una adopción más amplia fuera de InfluxDB, pero promete ser más poderoso y expresivo que InfluxQL, al llevar la carga de trabajo del análisis de series de tiempo a la base de datos. Sin embargo, el soporte para clustering de InfluxDB solamente se encuentra disponible en la versión empresarial, lo que ha limitado su adopción en algunos de nuestros proyectos.

Istio

PROBAR

Istio se está convirtiendo en la infraestructura por defecto para operar un ecosistema de microservicios. Su implementación fuera de la caja de varias preocupaciones cross-cutting como -descubrimiento de servicios, servicio a servicio, seguridad origen a servicio, observabilidad (incluyendo telemetría y trazado distribuido), rolling releases y resilencia - ha logrado recuperar en forma muy rápida nuestras implementaciones de microservicios. Es la principal técnica de implementación de la malla de servicio que hemos venido utilizando. Hemos disfrutado estas liberaciones mensuales y sus mejoras continuas con upgrades sin interrupciones. Utilizamos lstio para arrancar nuestros proyectos, comenzando con la observación (trazado y telemetría) y la seguridad de servicio a servicio. Estamos mirando de cerca sus mejoras de autentificación de servicio a servicio tanto dentro como fuera de la malla. Nos gustaría además ver que lstio establezca las mejoras prácticas para que la configuración de archivos logre un balance entre autonomìa a los desarrolladores del servicio y control a los operadores de la malla del servicio.

Kafka Streams

PROBAR

Kafka Streams es una librería pequeña para construir aplicaciones de streaming. Soporta APIs básicas de streaming como uniones, filtros, mapas y agregaciones así como almacenamiento local para casos de uso comunes como ventanas y sesiones. A diferencia de otras plataformas de procesamiento de streams como Apache Spark y Alpakka Kafka, Kafka Streams se ha ajustado muy bien a escenarios que no requieren una distribución en gran escala ni procesamiento en paralelo. Por lo tanto se lo puede utilizar sin ninguna otra pieza de infraestructura como cluster schedulers. Naturalmente, Kafka Streams ha sido una buena opción cuando se trabaja con el ecosistema de Kafka. Kafka Streams es particularmente útil cuando debemos procesar datos en un orden estricto y de forma exacta. Un caso de uso particular de Kafka Streams es para construir una plataforma (CDC) change data capture.

Nomad

PROBAR

HashiCorp sigue lanzando software interesante. Incluimos HashiCorp Vault en Marzo de 2017, y en esta edición del Radar hemos añadido otras tantas herramientas relacionadas con Terraform. Hemos movido Nomad a Trial porque hemos tenido experiencias positivas al utilizarlo. Mientras Kubernetes continúa ganando tracción, nos gusta la aplicabilidad general de Nomad. No se limita tan solo a ejecutar cargas de trabajo en contenedores, sino que puede utilizarse para planificar casi cualquier cosa. Soporta de manera nativa Java y Golang así como trabajos en lotes y trabajos de cron distribuidos. Nos gusta su focalización en operaciones sobre nubes múltiples e híbridas, algo que probablemente se va a volver más importante para evitar quedar cautivos de una sola ('sticky clouds') y el hecho de que es muy bueno en la planificación de tareas.

CloudEvents

EVALUAR

Fuera del propio código de la función, las aplicaciones escritas como funciones sin servidores (serverless) están fuertemente acopladas con la plataforma de la nube en la que se alojan. Aunque los eventos son un mecanismo usual para ejecutar las FaaS, y todos los proveedores de servicios en la nube los soportan de alguna manera, las especificaciones propietarias actuales impiden su interoperatividad entre nubes. La especificación CloudEvents es un estándar creciente que ha sido aceptado por la CNCF en su CNCF Sandbox. El estándar está siendo aún activamente desarrollado, pero ya existen implementaciones para varios lenguajes y Microsoft ha anunciado que dará soporte

prioritario en Azure. Esperamos que otros proveedores de servicios en la nube sigan su ejemplo.

Cloudflare Workers

EVALUAR

La mayoría de plataformas de ejecución de código server-side o serverless se centran alrededor de contenedores o VMs. Cloudflare Workers, sin embargo, toma un enfoque diferente para ofrecer hosting de computación serverless. Usa V8 Isolates, el motor de código abierto de JavaScript desarrollado para Chrome, para ejecutar funciones como servicio (functions as a service - FaaS) en su red de CDN extensiva. El código puede ser escrito en JavaScript o cualquier lenguaje que se compile a WebAssembly y los datos pueden ser accedidos desde la caché de Cloudflare o un repositorio clave-valor. El mayor beneficio para los desarrolladores es el rendimiento: al estar al límite de la red, cerca de los usuarios finales, los inicios en frío solo tardan cinco milisegundos. Para el proveedor, los beneficios incluyen ambos, la habilidad de empaquetar aislamientos densamente debido a la menor necesidad de memoria y mejor rendimiento al reducir los cambios de contexto entre procesos. Este es un enfoque definitivamente intrigante para monitorear y evaluar.

Deno

EVALUAR

Como grupo tenemos sentimientos entremezclados con respecto a programar en JavaScript en el lado del servidor, especialmente cuando la razón para hacerlo es evitar la programación políglota. Habiendo dicho esto, si decides usar JavaScript o TypeScript en el servidor, échale un vistazo a Deno. Escrito por Ryan Dahl, el inventor de Node.js, Deno intenta evitar lo que Ryan considera que son los errores que se cometieron al crear Node.js. Proporciona un entorno de pruebas estricto y un gestor integrado de dependencias y paquetes, y soporta TypeScript de forma natural. Deno se ha construido usando Rust y V8.

Hot Chocolate

EVALUAR

Tanto el ecosistema GraphQL como su comunidad siguen creciendo. Hot Chocolate es un servidor GraphQL para .NET (core y clásico). Permite construir y almacenar esquemas y realizar consultas contra estos. El equipo detrás de Hot Chocolate ha añadido recientemente costuras de esquema que permite con un único punto de entrada, consultar múltiples esquemas agregados desde diferentes ubicaciones. Aunque existen muchas formas de hacer mal uso de este enfoque, vale la pena evaluar si añadirlo o no a tu conjunto de herramientas.

Knative

EVALUAR

La arquitectura severless ha popularizado un estilo de programación FaaS entre los desarrolladores; este estilo los ayuda a enfocarse en resolver los problemas centrales del negocio con funciones independientemente construidas y desplegadas que reaccionan a un evento, ejecutan un proceso de negocio, producen otros eventos en el proceso y se reducen a cero. Históricamente, las plataformas serverless propietarias como AWS Lambda o Azure Functions de Microsoft han permitido desarrollar este paradigma de programación. Knative es una plataforma FaaS de código abierto basada en Kubernetes. Hay algunas cosas que resaltan acerca de Knative: es de código abierto y agnóstica al proveedor; implementa el flujo serverless como es descrito en la especificación del CNFC Serverless Working Group whitepaper; asegura interoperabilidad transversal a los servicios al implementar su interfaz de eventos consistente con la especificación CNCF CloudEvents; y, lo más importante, aborda el desafío común de operar un FaaS armónico y a la vez híbrido y una arquitectura basada en contenedores de larga ejecución. Se integra fácilmente con Istio y Kubernetes. Por ejemplo, los desarrolladores pueden utilizar estrategias de despliegue que Istio implementa dividiendo el tráfico entre diferentes versiones de las funciones. También pueden beneficiarse de la observabilidad provista por Istio, no sólo para servicios de contenedores de larga ejecución sino además para programas FaaS en el mismo entorno Kubernetes. Anticipamos que que la interfaz de eventos de Knative continuará habilitando nuevas integraciones de eventos de fuente y destino subyacentes.

MinIO

EVALUAR

El almacenamiento de objetos es una elección popular para almacenar datos no estructurados y, en algunos pocos casos, datos estructurados en la nube. Desaconsejamos el uso de la nube genérica (generic cloud), pero en el caso de querer minimizar el riesgo de cautividad de una nube (cloud stickiness) en lo que respecta al almacenamiento de objetos, MinIO nos ha parecido muy útil. Con una capa API compatible con S3, MinIO abstrae el almacenaje de objetos de los proveedores de la nube, incluyendo AWS, Azure y Google Cloud Platform (GCP), y lo hemos usado con éxito en productos con infraestructuras flexibles y con un propósito en concreto, desde centros de proceso de datos a proveedores de la nube.

PLATAFORMAS

si decides usar JavaScript o TypeScript en el servidor, échale un vistazo a Deno, que proporciona un entorno de pruebas estricto y un gestor integrado de dependencias y paquetes, y soporta TypeScript de forma natural

(Deno)

La arquitectura severless ha popularizado el estilo de programación "function-as-a-service" entre los desarrolladores. Knative es una plataforma de código abierto, agnóstica al proveedor. Plataforma basada en Kubernetes para ejecutar cargas de trabajo de función-como-un-servicio.

(Knative)

PLATFORMAS

Tendermint es un motor de replicación de una máquina de estados Byzantine que permite implementar tus propios sistemas de blockchain.

(Tendermint)

Prophet
EVALUAR

Incluso en el área de deep learning, los modelos estadísticos aún juegan un rol al apoyar la toma de decisiones de negocio. Los modelos de series de tiempo son ampliamente usados para pronosticar inventarios, demanda, tráfico de consumidores, etc. Crear estos modelos a mano tal que sean robustos y flexibles ha sido típicamente el rol de estadísticos especializados o de grandes proveedores de software comercial. Prophet es una alternativa de código abierto a paquetes comerciales de pronóstico que puede ser programado en R o Python. Facebook afirma que utiliza Prophet internamente para pronósticos de negocio a gran escala y lo ha disponibilizado como un paquete de código abierto para que cualquiera lo use. Nos gusta que Prophet elimine la tediosidad de la construcción de modelos, el mantenimiento y la manipulación de datos para que los analistas humanos y los expertos en la materia puedan centrarse en hacer lo que mejor saben hacer.

Quorum
EVALUAR

Quorum es "una versión enfocada a las empresas de Ethereum" que busca proveer control sobre permisos de red y privacidad de transacciones así como mayor rendimiento. Uno de nuestros equipos ha trabajado profundamente con Quorum; sin embargo, su experiencia hasta aquí no ha sido tan buena. Algunos desafíos resultan del uso con complex smart contract programming y otros del mismo Quorum. Por ejemplo. No trabaja bien con balanceadores de carga y solo tiene soporte parcial para base de datos, lo que lleva a una carga significativa en el despliegue. Nos encontramos con algunos problemas en la estabilidad y compatibilidad especialmente en transacciones privadas. Recientemente Quorum ha atraído bastante atención debido a JPM Coin. Sin embargo, desde una perspectiva técnica, recomendamos ser cautelosos a la hora de implementar Quorum y que se vigile su desarrollo.

SPIFFE
EVALUAR

La estandarización de servicio de identidad de SPIFFE ha sido un importante paso en la habilitación de soluciones turnkey para la encriptación end-to-end y autenticación mutua entre servicios. Los estándares SPIFFE están respaldados por la OSS SPIFFE Runtime Environment (SPIRE) que automáticamente entregan identidades criptográficamente demostrables a servicios. Istio tambien usa SPIFFE por defecto. SPIFFE habilita varios cosas de uso, incluyendo traducción de identidades, Autenticación de cliente OAuth, mTLS "cifrado en todas partes" y observabilidad de carga de trabajo. ThoughtWorks está trabajando activamente con las comunidades de Istio y SPIFFE para disminuir la brecha entre proveedores legados de servicios de identidad e identidades basadas en SPIFFE de tal manera que mTLS pueda ser usado en todas partes entre servicios, dentro y fuera de un service mesh.

Tendermint
EVALUAR

La tolerancia a fallos Bizantina (Byzantine fault tolerance, BFT) es uno de los problemas fundamentales de los sistemas de criptomonedas y blockchain. Requiere un acuerdo a nivel global del sistema a cerca de un valor único, en presencia de un número arbitrario de procesos que se comportan de manera defectuosa, entre los que se incluye el fraude malicioso. Tendermint es un motor de replicación de una máquina de estados BFT que permite implementar tus propios sistemas de blockchain. El motor de consenso, Tendermint Core, se encarga de la comunicación peer-to-peer y la parte del consenso, de modo que solo se tiene que implementar el resto de la aplicación (por ejemplo, construir la transacción y verificar la firma criptográfica) y comunicarse con Tendermint Core usando ABCI. Algunas implementaciones de blockchain ya han elegido Tendermint como su motor de consenso.

TimescaleDB
EVALUAR

En radares previos hemos discutido sobre PostgreSQL for NoSQL. La madurez de PostgreSQL y su extensabilidad ha llevado a un flujo constante de innovadores almacenes de persistencia construidos sobre el motor de Postgres. Uno que llamó nuestra atención fue TimescaleDB, una base de datos que permite escritura rápida y consultas optimizadas sobre datos en series de tiempo. Aunque (todavía) no tan completo como InfluxDB, TimescaleDB ofrece una alternativa de modelo de datos y capacidad de consultas. Se debería evaluar TimescaleDB si se tiene una necesidad modesta de escalamiento, se prefiere usar SQL y se aprecia la estabilidad y familiaridad de la interfaz administrativa que postgreSQL ofrece.

HERRAMIENTAS

Cypress
ADOPTAR

Seguimos recibiendo feedback positivo sobre herramientas "post-Selenium" de pruebas Web UI como Cypress, TestCafe and Puppeteer. Correr pruebas End-to-End puede presentar desafíos como un largo proceso de ejecución, poca fiabilidad de las pruebas y los desafíos de arreglar los fallos del CI cuando las pruebas se ejecutan en modo "headless". Nuestros equipos han tenido muy buenas experiencias con Cypress resolviendo cuestiones comunes como la falta de rendimiento y el largo tiempo de espera para las respuestas y durante la carga de los recursos. Cypress se ha convertido en la herramienta de elección dentro de nuestros equipos para realizar pruebas E2E.

Jupyter
ADOPTAR

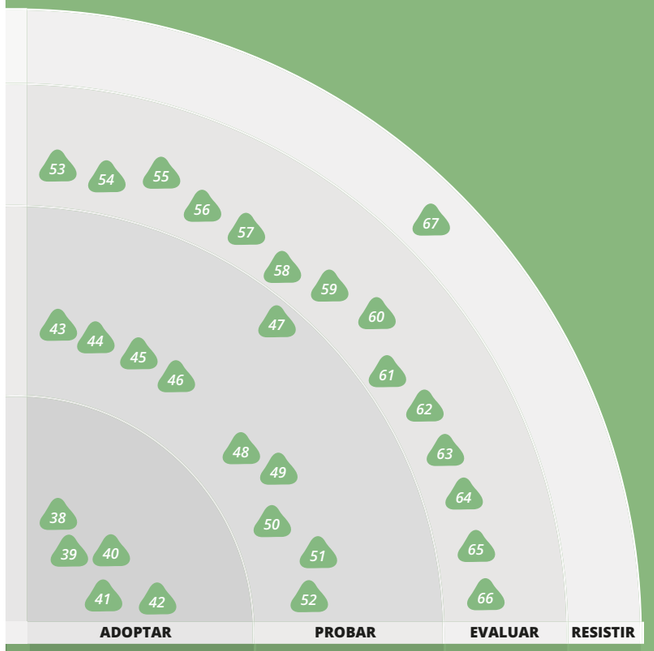
En los últimos años hemos notado una auge en la popularidad de los cuadernos analíticos. Éstos son aplicaciones inspiradas en Mathematica que combinan texto, visualización y código en un documento computacional vivo. Los Cuadernos Jupyter on utilizados por nuestros equipos para prototipado y exploración en análisis y aprendizaje automático. Hemos trasladado Jupyter a Adopt para esta edición del Radar para reflejar que ha salido ganador entre los cuadernos Python. Sin embargo, recomendamos cuidado especial al utilizar Cuadernos Jupyter en producción.

LocalStack
ADOPTAR

Uno de los retos de usar servicios en la nube es desarrollar y probar localmente. LocalStack resuelve esto para AWS proveyendo test doubles locales para una amplia variedad de servicios AWS incluyendo S3, Kinesis, DynamoDB y Lambda. Tiene fundamento en las mejores herramientas como Kinesalite, dynalite y Moto, agregando a esto la posibilidad de tener procesos aislados y funcionalidades de inyección de errores. LocalStack es muy simple de usar, ya que viene con JUnit ejecutable y una extensión de JUnit 5, además puede ser ejecutado dentro de un contenedor Docker. Hoy en dia, para muchos equipos se ha vuelto la herramienta por defecto de pruebas de servicios AWS.

Terraform
ADOPTAR

Terraform rápidamente se esta convirtiendo en una opción de facto para crear y gestionar infraestructuras en la nube mediante la escritura de definiciones declarativas. Usualmente se deja a Puppet, Chef o Ansible la configuración de los servidores instanciados por Terraform. Nos gusta Terraform porque la sintaxis de sus archivos es bastante legible, y también porque soporta varios proveedores de nube sin hacer ningún intento de proveer una abstracción artificial a través de los mismos. La comunidad activamente irá



ADOPTAR
38. Cypress
39. Jupyter
40. LocalStack
41. Terraform
42. UI dev environments

PROBAR
43. AnyStatus
44. AVA
45. batect
46. Elasticsearch LTR
47. Helm
48. InSpec
49. Lottie
50. Stolon
51. TestCafe
52. Traefik

EVALUAR
53. Anka
54. Cage
55. Cilium
56. Detekt
57. Flagr
58. Gremlin
59. Honeycomb
60. Humio
61. Operadores Kubernetes
62. OpenAPM
63. Systems
64. Taurus
65. Terraform provider GoCD
66. Terratest

RESISTIR
67. Handwritten CloudFormation

HERRAMIENTAS

Storybook, React Styleguidist, Compositor y MDX proveen un extenso entorno para iterar rápidamente sobre componentes de UI, haciendo foco en la colaboración entre desarrolladores y diseñadores de experiencia de usuario.

(Entornos de desarrollo de UI)

Malgastar mucha energía y esfuerzo en configurar ambientes de desarrollo local tratando de resolver el problema del "funciona en mi máquina local". batect permite configurar y compartir de forma fácil un ambiente de desarrollo basado en Docker, iniciando contenedores que ejecutan estas tareas.

(batect)

agregando el soporte para las últimas funcionalidades de la mayoría de los proveedores de nube. Siguiendo con nuestra primera y más cautelosa mención a Terraform de hace aproximadamente dos años, se ha visto un desarrollo continuo y ha evolucionado a un producto estable con un buen ecosistema que ha demostrado su valor en nuestros proyectos. El problema con el manejo de archivos de estado ahora puede ser eludido usando lo que Terraform llama "backend de estado remoto". Hemos usado con éxito AWS S3 para ese propósito.

Entornos de desarrollo de UI

ADOPTAR

A medida que más equipos adoptan DesignOps, sus prácticas y herramientas asociadas van madurando. Los entornos de desarrollo de UI proveen un extenso entorno para iterar rápidamente sobre componentes de UI, haciendo foco en la colaboración entre desarrolladores y diseñadores de experiencia de usuario. Hay varias opciones disponibles en este ámbito: Storybook, React Styleguidist, Compositor y MDX. Puedes usar estas herramientas separadas desarrollando componentes para librerías o de diseño de sistemas, así como embebidas en el desarrollo de una aplicación web. Muchos equipos fueron capaces de disminuir los ciclos de retroalimentación de UI, y mejorar el tiempo de trabajo de UI en preparación para el desarrollo. Todo esto ha hecho que tenga sentido para nosotros que "los entornos de desarrollo de UI" sea un práctica por defecto razonable.

AnyStatus

PROBAR

Ya que los desarrolladores usualmente enviamos muchos commits pequeños a diario, confiamos en monitores que nos indiquen cuando las nuevas compilaciones llegan a verde. AnyStatus es una aplicación ligera de Windows que elabora métricas y eventos de varias fuentes y las coloca en un solo sitio. Por ejemplo, resultados de la compilación y de lanzamientos, chequeos de salud para distintos servicios y métricas del SO. Imagínate que es CCTray en esteroides. También está disponible como plugin de Visual Studio.

AVA

PROBAR

AVA es un ejecutor de pruebas para Node.js. Aunque JavaScript es de hilo único (single-threaded), IO en Node. js puede ejecutarse en paralelo debido a su naturaleza asíncrona. AVA toma ventaja de esto y ejecuta tus pruebas concurrentemente, lo cual beneficia especialmente a tests IO pesados. Adicionalmente los archivos de pruebas se ejecutan en paralelo como procesos separados, dando como resultado mejor desempeño y un ambiente aislado para cada archivo. AVA es una opción liviana cuando se compara con frameworks completamente funcionales como Jest, por lo tanto se opone a esto y te fuerza a escribir casos de prueba atómicos.

batect

PROBAR

Se malgasta mucha energía y esfuerzo en configurar ambientes de desarrollo local tratando de resolver el problema del "funciona en mi máquina local". Por muchos años nuestros equipos han adoptado el enfoque "Check out and Go", en el cual usamos una solución basada en script para asegurar que nuestro ambiente local sea configurado consistentemente. batect es una herramienta de código abierto desarrollada por un ThoughtWorker, que permite configurar y compartir de forma fácil un ambiente de desarrollo basado en Docker. Batect se convierte en el script de entrada para el sistema de trabajo, iniciando contenedores que ejecutan estas tareas sin tener que depender de una configuración local. Los cambios en configuración y dependencias son compartidos de forma simple a través de un controlador de versiones sin requerir ningún cambio o instalación en máquinas locales o agentes CI. Si bien nos gusta Cage por sobre otras herramientas, vemos como Batect crece rápidamente en este ámbito en favor de nuestros equipos.

Elasticsearch LTR

PROBAR

Uno de los retos de realizar búsquedas es asegurar que los resultados más relevantes para el usuario aparezcan al inicio de una lista. Aquí es donde "learning to rank" (LTR - aprendiendo a clasificar) puede ayudar. LTR es el proceso de aplicar machine learning

para clasificar documentos obtenidos por un motor de búsqueda. Si tu estás usando Elasticsearch, puedes obtener una clasificación de búsqueda relevante con el plugin Elasticsearch LTR. Este plugin utiliza RankLib para generar los modelos durante la fase de entrenamiento. Entonces, cuando consultas a Elasticsearch, tu puedes usar este plugin para "re-puntuar" o "re-clasificar" los primeros resultados. Nosotros lo hemos usado en algunos proyectos y estuvimos felices con los resultados. También hay una solución LTR equivalente para usuarios de Solr.

Helm
PROBAR

Helm es un gestor de paquetes para Kubernetes. Viene con un repositorio de aplicaciones Kubernetes que son mantenidas en el repositorio oficial Charts. Helm tiene dos componentes: una utilidad para la línea de comandos llamada Helm y un componente de clúster llamado Tiller. Asegurar un clúster Kubernetes es un tema amplio, pero recomendamos configurar Tiller en un entorno de control de acceso basado en roles (RBAC). Hemos utilizado Helm en varios proyectos de clientes y su manejo de dependencias, plantillas y mecanismos de conexión han simplificado el ciclo de vida de las aplicaciones en Kubernetes. Sin embargo, recomendamos proceder con cautela, las plantillas YAML de Helm pueden ser difíciles de entender y Tiller todavía tiene algunos aspectos por resolver. Se espera que Helm 3 solucione estos temas.

InSpec
PROBAR

Cómo una organización da autonomía a los equipos de desarrollo asegurándose que las soluciones que despliegue sean seguras y cumplan con las regulaciones? Cómo asegurarse que los servidores, una vez desplegados, mantienen una configuración consistente? InSpec se posiciona como una solución para seguridad y compliance continuo, pero también puede usarse como infraestructura general para pruebas. InSpec permite la creación de pruebas de infraestructura declarativas, que pueden ser ejecutadas continuamente contra ambiente provistos, incluyendo producción. Particularmente, nuestros equipos elogian su diseño extensible con recursos y herramientas para múltiples plataformas. Recomendamos probar InSpec, como una solución al problema de garantizar seguridad y cumplimiento con las regulaciones.

Lottie
PROBAR

Una buena animación de interfaz de usuario puede mejorar bastante la experiencia de este. Sin embargo, para reproducir una delicada animación de diseñador en una app es usualmente una tarea desafiante para los desarrolladores. Lottie es una librería para Android, iOS, web, y Windows que parsea animaciones Adobe After Effects exportadas como JSON con Bodymovin y las representa nativamente en móvil y web. Tanto diseñadores como desarrolladores continuar utilizando sus herramientas familiares y colaborar fluidamente.

Stolon
PROBAR

Configurar instancias de alta disponibilidad de PostgreSQL puede llegar a ser complejo, por eso nos gusta Patroni, que nos ayuda a hacer más rápida la configuración de clusters PostgreSQL. Stolon es otra herramienta que hemos usado para configurar exitosamente clusters de instancias de PostgreSQL en producción usando Kubernetes. Aunque PostgreSQL soporta replicación por streaming desde el inicio, el reto en la configuración de alta disponibilidad está en asegurar que el cliente se conecte a la instancia principal siempre. Nos gusta que Stolon refuerza la conexión con la instancia principal de PostgreSQL cerrando las conexiones a otras instancias principales no activas y enrutando las peticiones a la instancia principal activa.

TestCafe
PROBAR

Tenemos buena experiencia utilizando herramientas de testing web "post-Selenium" como Cypress, TestCafe y Puppeteer. TestCafe te permite escribir tests en JavaScript o TypeScript y ejecuta tests dentro del propio navegador. TestCafe tiene varias funcionalidades útiles como ejecución paralela "out-of-the-box" y mocking de peticiones HTTP. TestCafe utiliza un modelo de ejecución asíncrono sin tiempos de espera explícitos, resultando en

HERRAMIENTAS

Confiar, pero verificar. InSpec ayuda garantizar que los servidores, una vez implementados, permanezcan seguros y compatibles a lo largo de su vida operativa.

(InSpec)

HERRAMIENTAS

Leveraging Linux eBPF, Cilium provee seguridad y redes compatibles con API-aware de una manera que se basa en la identidad del servicio, Pod o contenedor, y es dinámica y lista para microservicios.

(Cilium)

Detekt es una herramienta que realiza análisis estático de código para Kotlin. Esta herramienta encuentra code smells y complejidad de código. Se lo puede usar desde la consola o usando sus plugins de integración con herramientas de desarrollo populares.

(Detekt)

conjuntos de test mucho más estables. Su API de selección facilita la implementación de patrones PageObject. TestCafe recientemente ha lanzado la versión 1.0.x que mejora estabilidad y funcionamiento.

Traefik

EVALUAR

Traefik es un proxy inverso y balanceador de carga de código abierto. Si estás buscando un proxy de borde, que proporciona enrutamiento simple sin todas las características de NGINX y HAProxy, Traefik es una buena opción. El enrutador proporciona reconfiguración sin recarga, métricas, monitorización y el patrón disyuntor que son esenciales cuando se trabaja con microservicios. Además, se integra muy bien con Let's Encrypt para proporcionar la terminación SSL, también con componentes de infraestructura como Kubernetes, Docker Swarm o Amazon ECS para detectar automáticamente nuevos servicios o instancias para incluirlas en su balanceador de carga.

Anka

EVALUAR

Anka es un conjunto de herramientas para crear, gestionar y distribuir ambientes virtuales macOS que sean reproducibles para desarrollo iOS y macOs. Permite que se tenga una experiencia similar a la de docker en ambientes macOS: inicio instantáneo, una herramienta de línea de comando para gestionar máquinas virtuales, registrar la versión y etiquetar máquinas virtuales para su distribución. Descubrimos Anka cuando propusimos una solución macOS privada en la nube a un cliente. Vale la pena considerar esta herramienta cuando se aplica un flujo de trabajo de DevOps a ambientes iOS y macOS.

Cage

EVALUAR

Cage es una implementación de código abierto alrededor de Docker Compose que permite configurar y ejecutar múltiples componentes dependientes como una sola aplicación más grande. Permite orquestar la ejecución de componentes como imágenes de Docker, el código fuente del servicio desde el repositorio, los scripts para cargar almacenes de datos y pods que son contenedores que se ejecutan juntos como una unidad. Cage utiliza el formato de archivos de configuración Docker Compose v2. Soluciona algunos de los vacíos de Docker Compose, como el soporte de múltiples entornos, incluido el entorno dev para ejecutar una aplicación distribuida en la máquina del desarrollador local, el entorno de prueba para ejecutar pruebas de integración y producción.

Cilium

EVALUAR

El enfoque tradicional de seguridad de redes en Linux, así como sus iptables, es filtrar por direcciones IP y puertos TCP/UDP. Sin embargo, esas direcciones IP frecuentemente cambian en los dinámicos entornos de microservicios. Al aprovechar Linux eBPF, Cicilium proporciona seguridad y redes compatibles con APIs, al insertar la seguridad de forma transparente, basándose en la identidad del servicio, Pod o contenedor, a diferencia de la identificación de la dirección IP. Al desacoplar la seguridad del direccionamiento, Cicilium podría jugar un papel importante como una nueva capa de protección de red y les recomendamos que la revisen.

Detekt

EVALUAR

Detekt es una herramienta que realiza análisis estático de código para Kotlin. Esta herramienta encuentra code smells y complejidad de código. Se lo puede usar desde la consola o usando sus plugins de integración con herramientas de desarrollo populares como Gradle (para realizar análisis de código durante la compilación) o SonarQube (para realizar el análisis de cobertura de código además del análisis estático de código) e IntelliJ. Detekt es una gran adición para los pipelines de aplicaciones Kotlin.

Flagr

EVALUAR

El uso de feature toggles es una técnica importante en los escenarios de despliegue continuo. Hemos encontrado una serie de buenas soluciones locales, pero nos gusta

el enfoque por el cual Flagr permite definir feature toggles como servicio, y distribuirlos como un contenedor Docker. Viene con SDKs para los lenguajes principales, tiene una API REST simple y bien documentada y proporciona una cómoda interfaz.

Gremlin
EVALUAR

Gremlin es una SaaS que permite a organizaciones conducir experimentos de caos y así probar la resiliencia de sus sistemas. Viene con una serie de ataques de fallos — incluyendo recursos, red y fallos de estado — que pueden ser ejecutadas bajo demanda o de manera programada y requiere mínima configuración (especialmente para usuarios de Kubernetes, quienes pueden instalar Gremlin usando Helm). El cliente de Gremlin también posee una agradable interfaz web, lo que facilita la ejecución y administración de experimentos de caos.

Honeycomb
EVALUAR

Honeycomb es una herramienta de observabilidad que recibe datos importantes de los sistemas de producción y los hace manejables a través del muestreo dinámico. Las personas desarrolladoras pueden registrar grandes cantidades de importantes eventos y decidir luego como dividirlos y relacionarlos entre sí. Este enfoque interactivo es útil cuando se trabaja con los grandes sistemas distribuidos de hoy en día ya que hemos superados el punto en donde podemos anticipar razonablemente que preguntas podríamos hacerles a los sistemas productivos.

Humio
EVALUAR

Humio es una nueva herramienta en el terreno de administración de logs. Ha sido desarrollada desde el inicio para ser rápida tanto en el tratamiento de logs como en su consulta, usando un lenguaje incorporado sobre una base de datos time-series personalizada. Incorpora todo lo mejor del tratamiento, visualización y alarmas. En este aspecto Splunk y ELK Stack han dominado, así que tener una alternativa siempre es bueno. Estaremos observando el desarrollo de Humio con mucha atención.

Operadores Kubernetes
EVALUAR

Estamos muy entusiasmados por el impacto que Kubernetes ha tenido en la industria pero a la vez preocupados por la complejidad operacional que conlleva usarlo. Mantener un cluster Kubernetes en ejecución y administrar los paquetes que se despliegan en él requieren habilidades especiales y tiempo. Los procesos operacionales como actualizaciones, migraciones, respaldos, entre otros, pueden ser un trabajo de tiempo completo. Creemos que los Operadores Kubernetes jugarán un rol importante en reducir esta complejidad. El framework provee un mecanismo estándar para describir procesos operacionales automatizados para los paquetes que ejecutan en el cluster Kubernetes. Pese a que los operadores fueron inicialmente promovidos por RedHat, han empezado a emerger varios Operadores desarrollados por la comunidad para paquetes open-source populares como Jaeger, MongoDB y Redis.

OpenAPM
EVALUAR

Uno de los desafíos en adoptar una alternativa open-source frente a software comercial es sortear el complicado escenario de los proyectos para entender qué componentes se necesitan, cuáles juegan bien juntos y qué parte de la solución total cubre exactamente cada uno. Esto es particularmente difícil en el mundo de la observabilidad, donde la práctica estándar es comprar una solución completa pero costosa que haga todo. OpenAPM facilita la elección de herramientas para observabilidad open-source. Muestra las herramientas open-source clasificadas por rol de componentes, para que puedas seleccionar las que son compatibles interactivamente. Mientras mantengas la herramienta actualizada, debería ayudarte a navegar a través del confuso montón de posibilidades.

Systems
EVALUAR

Es fácil pensar en todos esos procesos en los que trabajamos como cadenas lineales de causa y efecto. La mayor parte del tiempo trabajamos con sistemas complejos en los que el ir y venir de feedback positivo y negativo influye en los resultados. Systems es una serie de herramientas que permite describir, ejecutar y visualizar diagramas de sistema. Usando un DSL compacto, ejecutandolo en Jupyter Notebook o a través de terminal, es fácil realizar la descripción de estos procesos y el flujo de información. Aunque es más una herramienta de nicho, puede ser interesante y divertido usarla.

HERRAMIENTAS

Humio es una nueva herramienta en el terreno de administración de logs. Ha sido desarrollada desde el inicio para ser rápida tanto en el tratamiento de logs como en su consulta, usando un lenguaje incorporado sobre una base de datos time-series personalizada.

(Humio)

Es una serie de herramientas que permite describir, ejecutar y visualizar sistemas complejos, en los que feedback positivo y negativo influye en los resultados

(Systems)

HERRAMIENTAS

Terraform provider GoCD te permite construir pipelines usando Terraform, una herramienta madura y ampliamente usada en el espacio de la infraestrucrtura como código. Tiene pruebas de regresión automatizadas para la API de GoCD, las cuales deberían ayudar a minimizar la cantidad de problemas al actualizar

(Terraform provider GoCD)

Taurus
EVALUAR

Taurus es una herramienta para evaluación de rendimiento de aplicaciones y servicios muy útil escrita en Python. Provee interfaces para varias herramientas de evaluación de rendimiento, incluyendo Gatling y Locust. Se lo puede ejecutar desde la consola e integrarlo fácilmente con flujos de trabajo de integración contínua para ejecutar pruebas de rendimiento en diferentes etapas del flujo de trabajo. Taurus también tiene buena reportería, ya sea como texto en la consola o integrado con una UI web interactiva. Nuestros equipos han hallado sencillo el configurar los archivos YAML de Taurus ya que se puede usar múltiples archivos para describir cada escenario de prueba y para referirse a todas las definiciones de cada escenario.

Terraform provider GoCD
EVALUAR

Terraform provider GoCD te permite construir pipelines usando Terraform, una herramienta madura y ampliamente usada en el espacio de la infraestrucrtura como código. Con esta herramienta puedes escribir pipelines en el Lengguaje HashiCorp de Configuración (HCL) capaces de usar todas las funciones de Terraform, incluyendo espacios de trabajo, módulos y estado remoto. Esta técnica es una excelente alternativa a Gomatic, que destacamos anteriormente en el blip de Pipelines como código. El SDK de Golang usado en este provider tiene pruebas de regresión automatizadas para la API de GoCD, las cuales deberían ayudar a minimizar la cantidad de problemas al actualizar.

Terratest
EVALUAR

Nosotros utilizamos ampliamente Terraform como código para configurar la infraestructura en la nube. Terratest es una librería escrita en Golang que hace más fácil escribir pruebas automatizadas para el código de infraestructura. La ejecución de una prueba crea componentes de infraestructura reales (como servidores, cortafuegos, o balanceadores de carga), despliega aplicaciones sobre ellos y valida que el comportamiento sea el esperado usando Terratest. Al finalizar el test, Terratest puede quitar la aplicación desplegada y limpiar los recursos. Esto la hace muy útil para probar de extremo-a-extremo tu infraestructura en un entorno real.

Handwritten CloudFormation
EVALUAR

AWS CloudFormation es un lenguaje declarativo propietario para aprovisionar la infraestructura AWS como código (Infrastructure as Code). Los archivos de CloudFormation suelen ser la propuesta predeterminada para la automatización de la infraestructura de AWS. Si bien esta podría ser una forma sensata de comenzar un proyecto pequeño, nuestros equipos y la industria en general han descubierto que Handwritten CloudFormation simplemente no escala a medida que crece la infraestructura. Las dificultades más notables de los archivos de CloudFormation para proyectos grandes incluyen poca capacidad de lectura, falta de declaraciones imperativas, limitada definición y uso de parámetros y falta de verificación de tipos. Abordar estas deficiencias ha generado un amplio ecosistema de herramientas de código abierto y personalizadas. Encontramos que Terraform es la herramienta por defecto que no solo aborda las deficiencias de CloudFormation sino que también tiene una comunidad activa para agregar las últimas características de AWS y corregir errores. Además de Terraform, se puede elegir entre muchas otras herramientas y lenguajes, como troposphere, sceptre, Stack Deployment Tool y Pulumi.

LENGUAJES & FRAMEWORKS

Apollo
ADOPTAR

Nuestros equipos informan que Apollo se ha convertido en la librería por excelencia al desarrollar una aplicación en React que usa GraphQL para obtener datos de un servicio back-end. A pesar de que el proyecto Apollo también provee un framework para el servidor y una pasarela (gateway) para GraphQL, el cliente Apollo llama nuestra atención porque simplifica el problema de conectar componentes de interfaz de usuario con datos servidos por cualquier backend GraphQL. En pocas palabras, esto significa menor cantidad de código que al usar REST backends y redux.

MockK
ADOPTAR

MockK es nuestra herramienta predilecta para hacer mocks cuando escribimos tests en aplicaciones Kotlin. Nos gusta usar esta librería por su soporte de primera clase para las características de Kotlin como coroutines o bloques lambda. Como librería nativa, ayuda a nuestros equipos a escribir código limpio y conciso en tests para aplicaciones Kotlin en vez de usar wrappers de Mockito o PowerMock.

TypeScript
ADOPTAR

TypeScript, un lenguaje estáticamente tipado y superconjunto de Javascript, se ha convertido en nuestro lenguaje por defecto. Proyectos de gran escala son los más beneficiados por la seguridad de tipos. Nuestros desarrolladores apoyan su gestión mínima de configuracion, buena integración en IDE y su capacidad de refactorizar código de forma segura y gradualmente adoptar tipos. Con el buen repositorio de definiciones de tipos de TypeScript a mano, nos podemos beneficiar de la riqueza de las librerías de JavaScript mientras ganamos seguridad de tipos.

Apache Beam
ADOPTAR

Apache Beam es un modelo de programación unificado open-source para definir y ejecutar pipelines de procesamiento de datos en paralelo en batch y streaming. El modelo Beam está basado en el modelo Dataflow que nos permite expresar lógica elegantemente de forma tal que podamos cambiar fácilmente entre batch, batch por ventanas o streaming. El ecosistema de procesamiento de big data ha estado evolucionando bastante, lo que hace difícil la elección del motor de procesamiento de datos correcto. Una de las razones clave para elegir Beam es que nos permite cambiar entre diferentes ejecutores. Algunos meses atrás, Apache Samza fue agregado a los ejecutores que ya soporta, los que incluyen Apache Spark, Apache Flink y Google Cloud Dataflow. Los diferentes ejecutores tienen distintas capacidades y proveer una API portable es una tarea difícil. Beam

ADOPTAR
68. Apollo
69. MockK
70. TypeScript

PROBAR
71. Apache Beam
72. Formik
73. HiveRunner
74. joi
75. Ktor
76. Laconia
77. Puppeteer
78. Reactor
79. Resilience4j
80. Room
81. Rust
82. WebFlux

EVALUAR
83. Aeron
84. Arrow
85. Chaos Toolkit
86. Dask
87. Embark
88. fastai
89. http4k
90. Immer
91. Karate
92. Micronaut
93. Next.js
94. Pose
95. react-testing-library
96. ReasonML
97. Taiko
98. Vapor

RESISTIR

LENGUAJES & FRAMEWORKS

joi es un lenguaje de descripción de esquemas y un validador de objetos JavaScript, independiente de cualquier framework web.

(joi)

Los sistemas reactivos vienen con escalabilidad y resiliencia mejorada, así como también, una mayor dificultad de depuración y una curva de aprendizaje más pronunciada. Algunos de nuestros proyectos han observado mejoras significativas en escalabilidad una vez que han sido movidos a Reactor y sus tecnologías Reactivas.

(Reactor)

intenta establecer un balance al incorporar activamente innovaciones de estos ejecutores en el modelo Beam y también al trabajar con la comunidad para influenciar el roadmap de estos ejecutores. Beam tiene SDKs en múltiples lenguajes incluyendo Java, Python y Golang. También hemos tenido éxito usando Scio el cual provee un wrapper Scala para Beam.

Formik
PROBAR

Formik es un componente de orden superior (high-order component) que facilita enormemente la sorprendentemente verbosa y compleja tarea de manejar formularios en React. Localiza administración de estado, asiste con el envío y opcionalmente usa Yup para simplificar la validación de datos.

HiveRunner
PROBAR

HiveRunner es un framework open-source para escribir pruebas unitarias para queries de Apache Hadoop Hive basado en JUnit4. Al escribir analítica no trivial o pipelines de datos en Hive SQL, creemos que HiveRunner es un buen habilitador para escribir pruebas e incluso hacer TDD con SQL moderadamente complicado. HiveRunner te habilita para escribir Hive SQL como artefactos completamente probados.

joi
PROBAR

joi es un lenguaje de descripción de esquemas y un validador de objetos JavaScript. Nos gusta que joi sea independiente de cualquier framework web, por lo que nuestros equipos pueden usar los mismos esquemas en diferentes plataformas. También puede usar las librerías complementarias para generar documentación de Swagger en las API que validan solicitudes con esquemas joi.

Ktor
PROBAR

Kotlin ha demostrado que su valor va más allá del desarrollo de aplicaciones móviles. Nuestros equipos han tenido buenas experiencias con Ktor en crear microservicios y poner código en producción. Ktor es un framework que, a diferencia de otros frameworks web que soportan Kotlin, está escrito en Kotlin, usando funcionalidades del propio lenguaje como coroutines que permiten una implementación no bloqueante y asíncrona. La flexibilidad de incorporar diferentes herramientas para "logging", DI o un motor de plantillas — además de su arquitectura ligera — hacen de Ktor una opción interesante para crear servicios RESTful.

Laconia
PROBAR

Laconia es un framework para desarrollar funciones AWS Lambda en JavaScript. A medida que el interés y uso de la tecnología serverless aumenta, también lo ha hecho la complejidad de las aplicaciones construidas. Laconia es un framework liviano que remueve parte de las asperezas que encontramos. Usa inyección de dependencias para aislar el código de tu aplicación de APIs AWS de más bajo nivel y provee adaptadores para los diferentes eventos a los que tu aplicación pueda responder. Juega bien con el Serverless Framework en tiempo de despliegue. Nos gustan los frameworks pequeños y livianos y Laconia es justamente eso.

Puppeteer
PROBAR

Puppeteer, así como Cypress y TestCafe, es una de las herramientas de pruebas de interfaz de usuario (UI) web que está ganando reconocimiento en nuestros equipos. Puppeteer puede tener un control detallado sobre navegadores sin interfaz gráfica, obtener trazas de tiempo para diagnósticos de carga y mucho más. Nuestros equipos piensan que Puppeteer es estable, además de rápido y más flexible que otras alternativas basadas en WebDriver.

Reactor
PROBAR

En ediciones previas del Radar, hemos hablado de Reactor. Reactor ha ganado continuamente fuerza en nuestros proyectos. Con su ecosistema Spring, se ha convertido en la implementación dominante de Reactive Streams. Los sistemas reactivos vienen con escalabilidad y resiliencia mejorada, así como también, una mayor dificultad de depuración y una curva de aprendizaje más pronunciada. Para aquellos proyectos en los que esto

es aceptable, Reactor ha probado ser una buena opción. Algunos de nuestros proyectos han observado mejoras significativas en escalabilidad una vez que han sido movidos a Reactor y sus tecnologías reactivas. Con R2DBC, se comenzó a tener soporte reactivo para controladores RDBMS los cuales son una de las debilidades de los servicios reactivos.

Resilience4j
PROBAR

Resilience4j es una librería ligera de tolerancia a fallos inspirada en Hystrix de Netflix. Nos gusta su estructura ligera y modular con la podemos incorporar módulos específicos para funcionalidades específicas como circuit-breaking, limitación de velocidad, reintentos y fuselaje. Mientras las mallas de servicios están tomando algunas de las capacidades de tolerancia a fallos, las librerías de tolerancia a fallos aún son un componente clave de nuestros sistemas cuando se trata de comportamiento de tolerancia a fallos para matices específicos a dominios y para servicios no contenerizados. Ya que Hystrix se encuentra en modo mantenimiento, Resilience4j se convierte en la elección por defecto en el ecosistema Java. Puede trabajar con APIs síncronas tal así como con reactivas. También proporciona métricas a dropwizard metrics, Prometheus etc usando módulos adicionales.

Room
EVALUAR

Room es una librería de persistencia para acceder a SQLite en Android. Hace el código de acceso a la base de datos más simple, con mínimo código redundante y más robusto, con verificación en tiempo de compilación de consultas SQL. Nuestros desarrolladores gustan de su completa integración con consultas observables usando LiveData. Room es uno de los componentes de Android Jetpack que fueron creados para que el desarrollo de aplicaciones

Rust
EVALUAR

Desde su última aparición en el Radar de enero del 2015, hemos visto un aumento sostenido en el interés en Rust. Algunos de nuestros clientes están utilizando Rust actualmente, principalmente en lo relativo a herramientas de infraestructura, pero también en dispositivos embebidos de gran potencia. El interés incrementó por un creciente ecosistema, así como también por las mejoras en el lenguaje en sí. Ésto último incluye mejoras en el rendimiento, pero también cambios que hacen a Rust más intuitivo, por ejemplo, el cambio hacia ámbitos no léxicos. La mayoría de los cambios se incluyen en la versión Standard del 2018, publicada en diciembre pasado.

WebFlux
EVALUAR

WebFlux es la implementación de Spring Framework para Reactive Streams. En general, vemos un aumento en los modelos de programación reactiva en nuestros equipos y del uso de WebFlux en equipos que trabajan en el ecosistema de Spring. Se utiliza de mejor forma en ecosistemas con microservicios grandes, donde el alto rendimiento de las peticiones es una preocupación importante. Permite el procesamiento de varias peticiones al mismo tiempo de forma asíncrona sin las complicaciones de usar muchos hilos. WebFlux usa Reactor como su biblioteca reactiva, pero es interoperable con otras librerías reactivas a través de Reactive Streams. Utiliza Netty como su motor de comunicaciones de alto rendimiento. Aunque fomentamos el uso de Reactive Streams, la adopción de este modelo de programación requiere un cambio significativo en el pensamiento.

Aeron
EVALUAR

Aeron es un transporte de mensajes peer-to-peer eficiente y confiable. Proporciona un registro de mensajes persistente y replicado a través de una serie de controladores que incluyen HTTP, UDP y TCP. También es compatible con la persistencia de almacenamiento streams de

LENGUAJES & FRAMEWORKS

Room hace el código de acceso a la base de datos de Android más simple, con mínimo código redundante y más robusto, con verificación en tiempo de compilación de consultas SQL

(Room)

LENGUAJES & FRAMEWORKS

Dask provee la manera de escalar Pandas, Scikit-Learn, y Numpy workflows con re-escritura mínima. Esto lo hace una opción interesante para científicos de datos que buscan escalabilidad horizontal para grandes conjuntos de datos.

(Dask)

fastai es una librería Python de código abierto que simplifica el entrenamiento rápido y preciso de redes neuronales posee soporte predefinido para visión computarizada, procesamiento del lenguaje natural (PLN) y más.

(fastai)

mensajes para su posterior reproducción. Para muchas aplicaciones, Aeron puede ser excesivo porque opera en un nivel bastante bajo (OSI Layer 4), pero su diseño peer-to-peer con su baja y predecible latencia son útiles en varios casos de uso. De hecho, hemos encontrado que es útil en ciertas aplicaciones de machine learning, así como en las arquitecturas basadas en eventos. Creemos que vale la pena señalar que existen protocolos de mensajería alternativos que no requieren servicios adicionales como Apache Kafka para ejecutarse.

Arrow

EVALUAR

Arrow es una librería de lenguaje funcional para Kotlin, creada al unir 2 populares librerías existentes (kategory y funKTionale). Mientras Kotlin provee bloques de construcción para programación funcional, Arrow proporciona un paquete de abstracciones de alto nivel listo para usar para desarrolladores Android. Proporciona tipos de datos, tipos de clases, efectos, ópticas y otros patrones de programación funcional así como integraciones con librerías populares. Con Arrow, las librerías existentes están unificadas, lo que a largo plazo debería evitar comunidades fraccionadas en este espacio.

Chaos Toolkit

EVALUAR

Chaos Toolkit es una de las herramientas de Chaos Engineering que aparecen en esta edición del Radar. Este conjunto de herramientas se usa para describir y después correr repetidos experimentos en tu infraestructura para entender su capacidad de recuperación tras un fallo. Muchos de nuestros equipos han estado usando herramientas propias para hacer esto, así que está bien ver un proyecto de código abierto dedicado a esta práctica. Este conjunto de herramientas ya contiene controladores para AWS, Azure Service Fabric y GCE (entre otras) y funciona bien con "build tools" lo que te permite experimentar con la automatización. Aún así, hay que ir con cuidado, Chaos Engineering es una técnica muy potente que es mejor utilizar en sistemas con capacidad de recuperación, es decir, sistemas construidos para soportar fallos. Por eso, nosotros recomendamos comenzar a usar Chaos Toolkit primero en un entorno que no sea el de producción.

Dask

EVALUAR

Los científicos e ingenieros de datos usualmente emplean librerías como la de pandas para realizar análisis de datos a medida, pero a pesar de ser expresivas y poderosas, estas librerías tienen una limitación crítica: trabajan sobre un solo CPU y no proveen escalabilidad horizontal para grandes conjuntos de datos. Dask, sin embargo, incluye un programador ligero de alto rendimiento que se puede escalar desde una laptop a un cluster de máquinas. Y dado que trabaja con NumPy, pandas y Scikit-learn, Dask parece prometedor para evaluaciones adicionales.

Embark

EVALUAR

En el pasado, para desarrollo de aplicaciones descentralizadas (dapp) hemos recomendado Truflle. Embark también puede hacer más fácil tu trabajo. Embark provee características como estructuración, construcción, prueba, depuración y la integración de almacenamientos descentralizados como IPFS. A través de su configuración declarativa, se puede administrar la configuración de contratos inteligentes, dependencias, artefactos y despliegue de forma muy sencilla. El tablero interactivo CLI de Embark también es impresionante. Todavía se ve personas utilizando Remix para escribir contratos y desplegar manualmente sus aplicaciones sin pruebas automatizadas, control de fuentes o administración de artefactos. Nos gustaría llamar la atención de las personas al desarrollo de aplicaciones descentralizadas, promoviendo herramientas como Truffle y Embark.

fastai

EVALUAR

fastai es una librería Python de código abierto que simplifica el entrenamiento rápido y preciso de redes neuronales. Está construida sobre PyTorch y se ha convertido en una popular herramienta para nuestros científicos de datos. fastai simplifica aspectos dolorosos del entrenamiento de modelos, como pre-procesamiento y carga de datos, a unas pocas líneas de código. Está construida sobre las mejores prácticas de aprendizaje

profundo y posee soporte predefinido para visión computarizada, procesamiento del lenguaje natural (PLN) y más. La motivación de sus fundadores ha sido la de crear una librería fácil de usar para aprendizaje profundo y una sucesora mejorada de Keras. GCP, AWS y Azure han adoptado rápidamente en sus imágenes de máquinas. Los creadores de fastai, conociendo la velocidad y limitaciones de seguridad de Python, han anunciado la adopción de Swift como un lenguaje alternativo para el aprendizaje profundo. Estaremos mirando de cerca este espacio.

http4k

EVALUAR

http4k es un conjunto de herramientas HTTP escrito en Kotlin puro, y se usa para servir y consumir servicios HTTP. Una de las ideas clave detrás de http4k es que las aplicaciones HTTP son modeladas al componer dos simples funciones: HttpHandler y Filter. Derivan de la inspiración de la publicación de Twitter "Your Server as a Function". Es muy liviana, donde el módulo principal no tiene otras dependencias aparte de la StdLib de Kotlin. Además de su elegancia y simplicidad, también nos gusta su énfasis en la facilidad de escribir pruebas. Dado que las entidades en las librerías son inmutables, y que las rutas en la aplicación, tal como la aplicación misma, son sólo funciones, son muy fáciles de probar. Una de las cosas que se debe tener cuidado es que aún no existe soporte para no-bloqueante y co-rutinas en http4k aún.

Immer

EVALUAR

Con la creciente complejidad de aplicaciones JavaScript single-page, manejar la predictibilidad del estado se ha vuelto cada vez más importante. La inmutabilidad puede ayudar a asegurar que nuestras aplicaciones se comporten de forma consistente, sin embargo, desafortunadamente JavaScript no soporta nativamente la habilidad de crear objetos inmutables. Las librerías como Immutable.js han llenado ese vacío pero a la vez introducido nuevos problemas, ya que ahora dos tipos de objetos y arreglos existen en la aplicación, los de la librería y los nativos de JavaScript. Immer, siempre en Alemán, es un pequeño paquete que te permite trabajar con estado inmutable en una forma más conveniente. Está basado en el mecanismo copia-en-escritura, tiene una API minimal y opera sobre objetos y arreglos JavaScript normales. Esto significa que el acceso a los datos es transparente y no se requieren grandes esfuerzos de refactoring al introducir inmutabilidad a la base de código.

Karate

EVALUAR

Dada nuestra experiencia sobre que los tests son la única especificación de API que realmente importa, siempre estamos en la búsqueda de nuevas herramientas que puedan ayudar. Karate es un framework para testing de APIs cuya característica única es que los tests se escriben directamente en Gherkin, sin la necesidad de requerir un lenguaje de programación de propósito general para implementar comportamiento de pruebas. Karate es realmente un lenguaje específico al dominio para describir tests de APIs basadas en HTTP. Pese a que el enfoque es interesante y logra conseguir especificaciones muy legibles para tests simples, el lenguaje de propósito especial para evaluar los resultados puede ser bastante pesado en la sintaxis y difícil de entender. Queda por verse si pruebas más complejas escritas con este estilo serán legibles y mantenibles en el tiempo.

Micronaut

EVALUAR

Micronaut es un nuevo framework de JVM para crear microservicios usando Java, Kotlin o Groovy. Destaca por su poco uso de memoria y corto tiempo de puesta en marcha. Consigue estas mejoras evitando repercusiones en la ejecución causadas por DI y generación de proxies, un problema común de otros frameworks, y en su lugar usa un contenedor DI/AOP que genera la inyección de dependencias durante la compilación. Esto le da atractivo no sólo para microservicios de servidor sino también en el contexto de, por ejemplo, Internet de las cosas, aplicaciones de Android y funciones sin servidor. Micronaut usa Netty y soporta la programación reactiva. Incluye también muchas funcionalidades que lo hacen apto para la nube como el descubrimiento de servicios y el corte de circuitos. Micronaut entra en el framework de la JVM siendo muy prometedor y lo iremos siguiendo con interés

LENGUAJES & FRAMEWORKS

http4k es una herramienta HTTP muy liviana escrito en Kotlin puro, y se usa para servir y consumir servicios HTTP. Es elegante y simple, con énfasis en la facilidad de escribir pruebas.

(http4k)

Micronaut es un nuevo framework de JVM para crear microservicios usando Java, Kotlin o Groovy. Destaca por su poco uso de memoria y corto tiempo de puesta en marcha.

(Micronaut)

LENGUAJES & FRAMEWORKS

Taiko es una librería de node.js con una API clara y concisa para ayudar en la automatización de navegadores chrome o chromium. Pruebas escritas en Taiko Tests written in Taiko están destinadas a ser altamente legibles y mantenibles.

(Taiko)

Next.js
EVALUAR

React.js ha revolucionado la forma en que la mayoría de personas escriben "single page applications" (SPA) en JavaScript. Generalmente, recomendamos utilizar App Create React en todo el ciclo de vida de la aplicación de manera que no se tenga que configurar, construir y empaquetar manualmente. Sin embargo, algunos desarrolladores prefieren una herramienta cuyos valores iniciales por defecto reflejan un conjunto de opiniones sólidas. Next.js es justamente un framework opinionado y está ganando gran interés entre nuestros entusiastas de front-end. Next.js simplifica el routing, se renderiza en el servidor por defecto y coordina la construcción y dependencias. Estamos ansiosos por ver si cumple con las expectativas en nuestros proyectos.

Pose
EVALUAR

Pose es una librería de animación parecida a CSS para React.js, React Native y Vue.js. Es un sistema de moción declarativa que combina la simplicidad de la sintaxis de CSS con la fuerza y la flexibilidad de las animaciones e interacciones de JavaScript.

react-testing-library
EVALUAR

Como el ritmo de cambio en los frameworks de JavaScript se ha ralentizado, nuestros equipos tienen más tiempo para trabajar con frameworks específicos y, como resultado, han ganado más conocimiento sobre estos. Con React y el framework de pruebas dominante, Enzyme, hemos observado una tendencia preocupante a que las pruebas unitarias estén excesivamente acopladas a los detalles de implementación. Como se centran en detalles, estas pruebas no nos dan mucha confianza sobre si las funcionalidades hacen lo que deben. Estas pruebas unitarias dificultan la evolución del diseño y dejan demasiada responsabilidad para la cúspide de la pirámide de pruebas, es decir a las pruebas funcionales.
Por esto hemos revisitado la idea de pruebas subcutáneas. Adicionalmente, por su diseño, Enzyme tiene dificultades para estar al día con el desarrollo de React. Por todos estos motivos estamos considerando react-testing-library como una nueva framework de pruebas para aplicaciones en React.

ReasonML
EVALUAR

ReasonML es un nuevo e interesante lenguaje basado en OCaml con tintes de sintaxis de C que usa JavaScript como destino de la compilación por defecto. Creado por Facebook, permite embeber fragmentos de código JavaScript y plantillas JSX con buena integración para React. Pretende ser accesible para desarrolladores de JavaScript y aprovecha su ecosistema, y a la vez provee tipado seguro en un lenguaje funcional.

Taiko
EVALUAR

Taiko es una librería de node.js con una API clara y concisa para ayudar en la automatización de navegadores Chrome o Chromium. Se puede aprovechar los selectores inteligentes de Taiko para escribir pruebas confiables a medida que evoluciona la estructura de la aplicación web. No hay necesidad de selectores por Id, CSS o XPath o de agregar tiempo de espera explícito (para solicitudes XHR) en los scripts de prueba. La grabadora interactiva de comandos es útil cuando se desea desarrollar las pruebas al mismo tiempo que se explora la funcionalidad. Aunque se puede usar Taiko de forma independiente, hemos tenido un gran éxito al usarlo con Gauge.

Vapor
ASSESS

Somos defensores de ser agnósticos al lenguaje de programación, pero reconocemos que en algunos casos tiene sentido enfocarse en un solo lenguaje. Si han invertido fuertemente en Swift, más que todo por el desarrollo en iOS, y están buscando una tecnología para escribir servicios del lado de servidor, echen una mirada a Vapor, un moderno framework web para Swift que ha ganado una considerable popularidad.

¿Quieres estar al día con todas las noticias y puntos de vista del Radar?
Síguenos en tu red social favorita
o conviértete en suscriptora/o.

suscríbete ahora

ThoughtWorks®

Somos una consultora de software y una comunidad de individuos apasionados guiados por propósitos. Pesamos de manera disruptiva para ofrecer tecnología para enfrentar los desafíos más difíciles de nuestros clientes, mientras intentamos revolucionar la industria de TI y crear un cambio social positivo.

Fundada hace 25 años, ThoughtWorks se ha convertido en una compañía de más de 6000 personas, incluyendo una división de productos que crea herramientas pioneras de software para equipos. ThoughtWorks cuenta con 40 oficinas en 14 países: Australia, Alemania, Brasil, Canada, Chile, China, Ecuador, España, Estados Unidos, India, Italia, Reino Unido , Singapur y Tailandia,

thoughtworks.com

ThoughtWorks®

thoughtworks.com/es/radar

#TWTechRadar