ThoughtWorks®

TECHNOLOGY RADAR VOL. 21

Una guía con opiniones sobre 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 San Francisco en Octubre 2019

Traducido por: Abel Guillén, Aldemaro Díaz, Alejandro Mesías, Alexandra Granda, Andrea Santacruz, Araceli Correa, Bárbara Deluchi, Byron Torres, Carlos Oquendo, Cecilia Barudi, Daniel Fratte, Daniela Cortés, Daniela Mora, Daniela Garcés, David Montaño, David Corrales, David Salazar, Eduard Maura i Puig, Felipe Talavera, Gabriela Guamán, Glenn Wolfschoon, Gonzalo Rodríguez, Iván Ortega, Javiera Laso, Jesús Cardenal, José Puebla, Lorena Campos, Luis Azcona, María Vilatuña, María José Lalama, Marcos Mercuri, Mariana Perez, Mateo Rojas, María Fernanda Yepez, Mónica Giraldo, Mónica Sanchez, Nahuel Sotelo, Pamela Guamán, , Rayner Pupo, Reynier Pupo, Ricardo Rodriguez, Tex Albuja, Vicky Valverde y Yago Pereiro.

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

Nuestros Thoughtworkers son 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, son quienes 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.

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 más información del Radar, entra 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

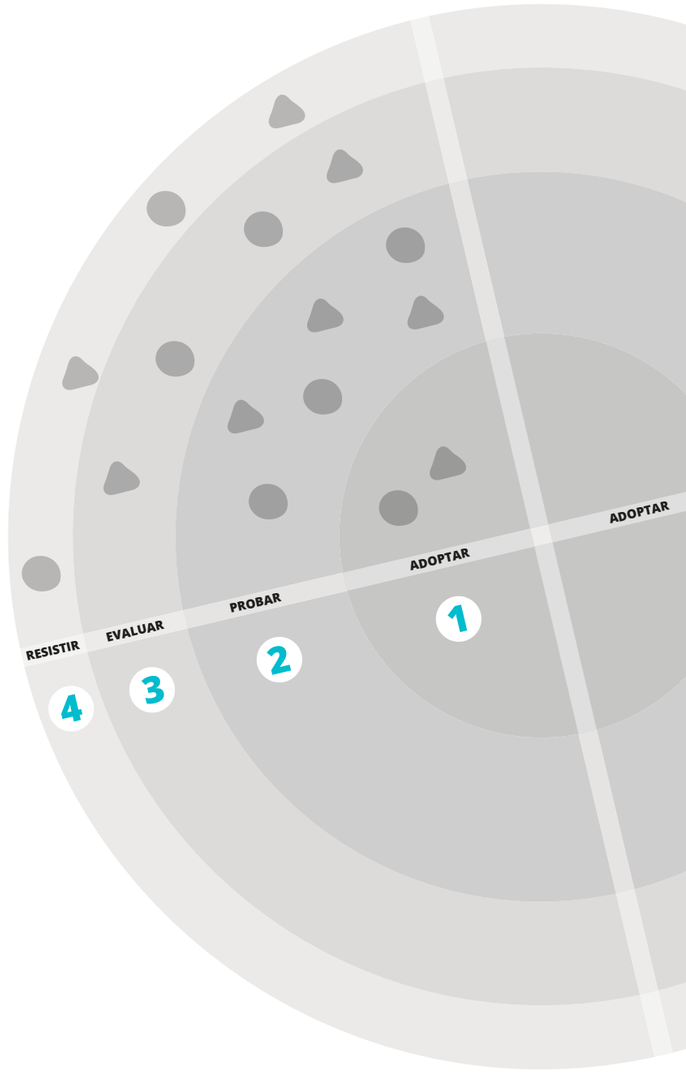
Vale la pena probarlos. Es importante entender cómo desarrollar estas capacidades. Las empresas deberían probar esta tecnología en proyectos en que se puede manejar el riesgo.

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

Cloud: ¿Más es menos?

A medida que los principales proveedores de la nube alcanzan paridad en las funcionalidades principales, el enfoque competitivo se ha trasladado a los servicios adicionales que pueden proporcionar, alentándolos a lanzar nuevas ofertas a una velocidad violenta. En su apuro por competir, vemos nuevos servicios en el mercado con funcionalidades sin pulir y características incompletas. El énfasis en la velocidad y la proliferación de productos, ya sea a través de la adquisición o de la creación apresurada, a menudo resulta no solo en errores, sino también en documentación deficiente, difícil automatización e integración incompleta entre las propias partes de los proveedores. Esto causa frustración para los equipos que intentan entregar software utilizando la funcionalidad prometida por el proveedor de la nube ya que se enfrentan constantemente a obstáculos. Las empresas eligen a los proveedores de la nube por una variedad de factores y, a menudo, a un alto nivel en la organización. Nuestro consejo para los equipos individuales es: no asumir que todos los servicios de su proveedor de nube designado son de la misma calidad, prueben las funcionalidades clave y estén abiertos a seleccionar alternativas de código abierto o una estrategia polycloud, si los beneficios ameritan la sobrecarga operativa.

Protegiendo la Cadena de Suministro de Software

Las organizaciones deberían de ser resistentes a las reglas de gobierno de torre de marfil, que requieren largas inspecciones de manuales y de aprobación; en vez, la protección de dependencias automatizada (Funciones de Diagnóstico de Dependencia), seguridad (Políticas de Seguridad como Código), y otros mecanismos de gobierno (Funciones de Diagnóstico de Arquitectura) protegen lo importante pero no lo urgente de los proyectos de software. Este tema sobre política, cumplimiento y gobierno como código ha aparecido una y otra vez en nuestras conversaciones. Vemos una evolución natural en el ecosistema del desarrollo de software de aumentar la automatización: integración continua con pruebas y comprobación automatizada, entregas continuas, infraestructura como código, y ahora gobernanza automatizada. Construir la automatización alrededor del coste de la nube, gestión de dependencias, estructura de arquitectura y otros antiguos procesos manuales demuestra una evolución natural; estamos aprendiendo cómo automatizar todos los aspectos importantes de la entrega de software.

¿QUÉ HAY DE NUEVO?

Temas destacados en esta edición

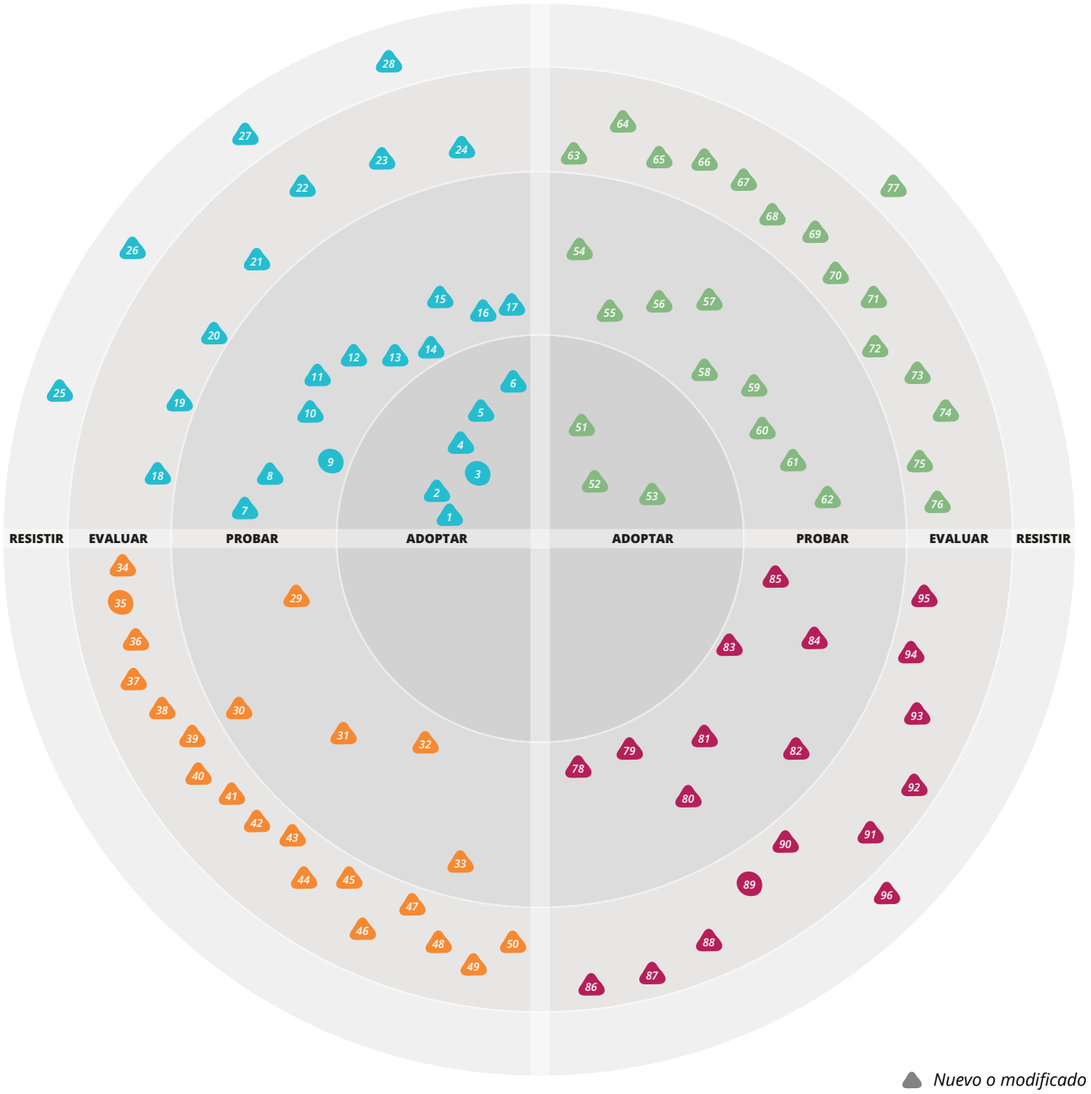
Interpretando el misterio que es Machine Learning

Machine learning a menudo parece descubrir soluciones a problemas que los humanos no pueden, utilizando búsqueda de patrones, retro-propagación y otras técnicas conocidas. Sin embargo, a pesar de su poder, muchos de estos modelos son inherentemente opacos, lo que significa que sus resultados no pueden explicarse en términos de inferencia lógica. Este es un problema cuando los humanos tienen derecho a saber cómo se tomó una decisión o cuando existe el riesgo de introducir sesgos de prejuicio, muestreo, algoritmo u otros en el modelo. Ahora estamos viendo la aparición de herramientas como What-If y técnicas como las pruebas de sesgo ético que nos ayudan a encontrar las limitaciones y predecir el resultado de estos modelos. Si bien estas mejoras en la interpretabilidad son un paso en la dirección correcta, explicar las redes neuronales profundas sigue siendo un objetivo difícil de alcanzar. Por esa razón, los científicos de datos están comenzando a considerar la explicabilidad como una principal necesidad al elegir un modelo de aprendizaje máquina.

Desarrollo de Software como Deporte en Equipo

Desde los primeros días de nuestro Radar tecnológico, hemos advertido acerca las herramientas y técnicas que aíslan a los miembros de los equipos de software, lo que dificulta la retroalimentación y la colaboración. A menudo, cuando aparecen nuevas especializaciones, los profesionales, los proveedores y las herramientas insisten en que parte del desarrollo debe realizarse en un entorno aislado, alejado del caos del desarrollo “regular”. Rechazamos esa afirmación y buscamos constantemente nuevas formas de abordar el desarrollo de software como un deporte de equipo. La retroalimentación es crítica al construir cosas complejas como el software. Si bien los proyectos requieren cada vez más especialización, nos esforzamos por adaptarlos a la colaboración y retroalimentación. Nos disgusta especialmente el meme de “10x engineers”, preferimos centrarnos en crear y habilitar “equipos 10x “. Vemos que esto se está desarrollando actualmente en cómo el diseño, la ciencia de datos y la seguridad pueden integrarse en equipos multifuncionales y apoyarse con una automatización sólida. La próxima frontera es traer más actividades de gobernanza y cumplimiento.

EL RADAR



TÉCNICAS

ADOPTAR
1. Análisis de seguridad en contenedores
2. Integridad de datos en el origen
3. Micro frontends
4. Pipelines para infraestructura como código
5. Coste de ejecución como función de aptitud de arquitectura
6. Pruebas utilizando dispositivos reales

PROBAR
7. Machine learning automatizado (AutoML)
8. Certificación Binaria
9. Entrega continua para modelos de machine learning (CD4ML)
10. Descubrimiento de datos
11. Función de aptitud para montículos de dependencias
12. Design systems
13. Herramientas de seguimiento para experimentación con machine learning
14. La explicabilidad como criterio-relevante al seleccionar un modelo
15. Política de seguridad como código
16. Sidecars para seguridad de endpoints
17. Zhong Tai

EVALUAR
18. BERT
19. Malla de datos
20. Pruebas de sesgo ético
21. Aprendizaje federado
22. JAMstack
23. Vincular registros conservando la privacidad (PPRL) con el filtro de Bloom
24. Ciclos de aprendizaje semi-supervisados

RESISTIR
25. 10x engineers
26. Integración de front-end a través de artefactos
27. Lambda pinball
28. Paridad de funcionalidades en migraciones legadas

PLATAFORMAS

ADOPTAR

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

EVALUAR
34. AWS Cloud Development Kit
35. Azure DevOps
36. Azure Pipelines
37. Crowdin
38. Crux
39. Delta Lake
40. Fission
41. FoundationDB
42. GraalVM
43. Hydra
44. Kuma
45. MicroK8s
46. Oculus Quest
47. ONNX
48. Contenedores sin raíces
49. Snowflake
50. Teleport

RESISTIR

HERRAMIENTAS

ADOPTAR
51. Commitizen
52. ESLint
53. React Styleguidist

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

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

RESISTIR
77. Azure Data Factory para orquestación

LENGUAJES & FRAMEWORKS

ADOPTAR

PROBAR
78. Arrow
79. Flutter
80. jest-when
81. Micronaut
82. React Hooks
83. Biblioteca de Pruebas React
84. Styled components
85. Tensorflow

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

RESISTIR
96. Enzyme

TÉCNICAS

Análisis de seguridad en contenedores

ADOPTAR

La adopción continua de contenedores para los despliegues, en particular Docker, ha tornado el análisis de seguridad de los contenedores en una técnica necesaria, y hemos movido esta técnica a “Adoptar” para reflejar este hecho. En particular, los contenedores introdujeron una nueva vía para los problemas de seguridad; es vital que se utilicen herramientas para analizar y revisar los contenedores durante el despliegue. Preferimos utilizar herramientas de análisis automático que corran como parte del pipeline de despliegue.

Integridad de datos en el origen

ADOPTAR

Hoy en día, la respuesta de muchas empresas para acceder a datos para su uso analítico es construir un laberinto complejo de pipelines de datos. Estos pipelines recogen los datos de una o múltiples fuentes, los limpian, los transforman y los mueven a otra ubicación para su consumo. Este acercamiento al manejo de datos muchas veces deja a los pipelines consumidores con la difícil tarea de verificar la integridad de los datos entrantes y construir una lógica compleja para limpiar esos datos y permitir que lleguen al nivel requerido de calidad. El problema fundamental con esto es que la fuente no tiene ningún incentivo ni responsabilidad a la hora de proveer datos de calidad para sus consumidores. Por esta razón, abogamos por la integridad de los datos en su origen, lo cual quiere decir que cualquier fuente que provea datos consumibles debe describir explícitamente sus medidas para lograr la calidad y garantizar aquellas medidas. La razón principal detrás de esto es que los sistemas y equipos de origen son los más familiarizados con sus datos y los que están en mejor posición para arreglarlos en la fuente. La arquitectura Malla de datos lleva esto un paso más allá, comparando los datos consumibles a un producto, donde la calidad de estos y sus objetivos son atributos integrales de cada colección de datos compartida.

Micro frontends

ADOPTAR

Hemos visto beneficios significativos al utilizar microservicios, estos han permitido a los equipos escalar la entrega de servicios desplegados y mantenidos de manera independiente. Desafortunadamente, hemos visto a equipos creando monolitos en el front-end — una gran aplicación web y enredada sobre servicios back-end — neutralizando en gran parte los beneficios de los microservicios. Los micro frontends continúan ganando popularidad desde que se presentaron por primera vez. Hemos visto a muchos equipos adoptar de alguna manera esta arquitectura para manejar la complejidad de múltiples desarrolladoras/es y equipos contribuyendo en la misma experiencia de usuario. En junio de este año, uno de los autores de esta técnica, publicó un artículo que sirve de referencia como un introductorio a micro frontends. Este muestra cómo se puede implementar este estilo utilizando varios mecanismos de programación web así como la construcción de una aplicación React.js. Estamos seguras/os en que éste estilo crecerá en popularidad a medida que las organizaciones descompongan el desarrollo UI en varios equipos.

ADOPTAR

1. Análisis de seguridad en contenedores
2. Integridad de datos en el origen
3. Micro frontends
4. Pipelines para infraestructura como código
5. Coste de ejecución como función de aptitud de arquitectura
6. Pruebas utilizando dispositivos reales

PROBAR

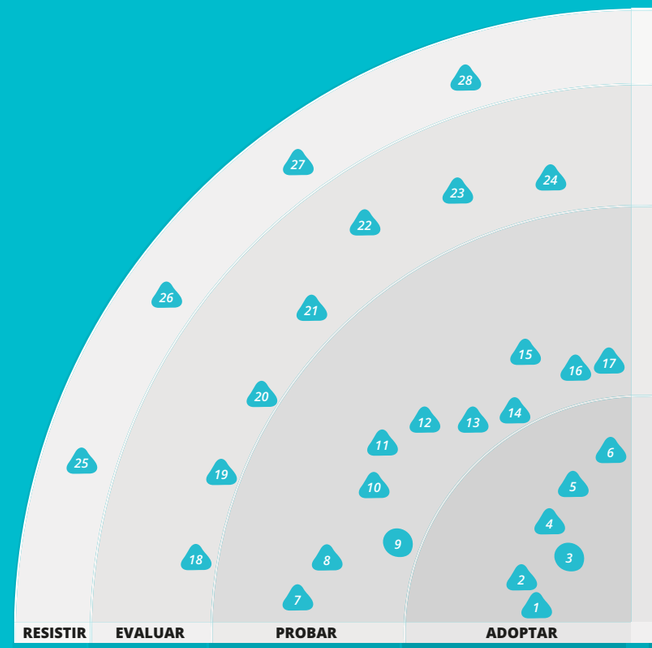
7. Machine learning automatizado (AutoML)
8. Certificación Binaria
9. Entrega continua para modelos de machine learning (CD4ML)
10. Descubrimiento de datos
11. Función de aptitud para montículos de dependencias
12. Design systems
13. Herramientas de seguimiento para experimentación con machine learning
14. La explicabilidad como criterio-relevante al seleccionar un modelo
15. Política de seguridad como código
16. Sidecars para seguridad de endpoints
17. Zhong Tai

EVALUAR

18. BERT
19. Malla de datos
20. Pruebas de sesgo ético
21. Aprendizaje federado
22. JAMstack
23. Vincular registros conservando la privacidad (PPRL) con el filtro de Bloom
24. Ciclos de aprendizaje semi-supervisados

RESISTIR

25. 10x engineers
26. Integración de front-end a través de artefactos
27. Lambda pinball
28. Paridad de funcionalidades en migraciones legadas



TÉCNICAS

Hemos visto emerger herramientas machine learning Automatizado (AutoML) que pretenden facilitar a personas sin expertise la automatización de inicio a fin del proceso de selección y entrenamiento de modelos.

(AutoML)

La certificación binaria (Binary attestation) es una técnica para el control de la seguridad al momento de despliegue.

(Certificación Binaria)

Pipelines para infraestructura como código

ADOPTAR

El uso de pipelines de entrega continua para orquestar el proceso de entrega de software se ha vuelto un concepto popular. Herramientas de CI/CD pueden ser usadas para testear configuraciones de servidores (ej: Packer), aprovisionamiento de ambientes (ej: Terraform, CloudFormation) y la integración de ambientes. El uso de pipelines para infraestructura como código permite encontrar errores antes de que los cambios sean aplicados en los entornos operacionales - incluyendo entornos utilizados para desarrollo y test. También ofrecen una manera de asegurar que las herramientas de infraestructuras se están ejecutando consistentemente, utilizando agentes CI/CD en vez de workstations individuales. Nuestros equipos han tenido buenos resultados utilizando esta técnica en sus proyectos.

Coste de ejecución como función de aptitud de arquitectura

ADOPTAR

Automatizar la estimación, seguimiento y proyección del coste de ejecución de una infraestructura en la nube es necesario para las organizaciones de hoy. Los modelos inteligentes de precios de los proveedores de la nube, combinados con la proliferación de los parámetros de precios y la naturaleza dinámica de la arquitectura de hoy pueden llevar a un costo de ejecución sorprendentemente alto. Por ejemplo, los precios de serverless basados en llamadas API, de soluciones de streaming de eventos enfocadas en el tráfico o de procesamientos de grupos de datos basados en tareas corridas, tienen una naturaleza dinámica que cambia a lo largo del tiempo a medida que la arquitectura evoluciona. Cuando nuestros equipos manejan infraestructura en la nube, implementar el coste de ejecución como función de la aptitud de arquitectura es una de sus primeras tareas. Esto quiere decir que nuestros equipos pueden observar el costo de ejecutar los servicios en comparación al valor entregado; cuando ven desviaciones respecto a lo que se espera o es aceptable, discutirán si es momento de evolucionar la arquitectura. La observación y cálculo del coste de ejecución se implementa como una función automatizada.

Pruebas utilizando dispositivos reales

ADOPTAR

Cuando se adopta con éxito la entrega continua (CD), los equipos se esfuerzan por hacer que sus diversos ambientes de prueba se vean lo más cercanos posible a la producción. Esto les permite evitar errores que sólo se podrían ver directamente en el ambiente de producción. Esto sigue siendo igual de válido para softwares incrustados y del Internet de las cosas (Internet of Things); si no ejecutamos las pruebas en ambientes reales, podríamos encontrar algunos errores por primera vez en producción. Las pruebas utilizando dispositivos reales ayudan a evitar este problema asegurando que se utilicen dispositivos reales en el pipeline de entrega continua.

Machine Learning Automatizado (AutoML)

PROBAR

La potencia y promesa de machine learning ha creado una demanda de experiencia que supera la oferta de científicos de datos especializados en este área. Como respuesta a esta brecha de habilidades, hemos visto emerger herramientas como machine learning Automatizado (AutoML), que pretenden facilitar a personas sin expertise, la automatización de inicio a fin del proceso de selección y entrenamiento de modelos. Algunos ejemplos incluyen AutoML de Google, DataRobot, y the H2O AutoML interface. Aunque hemos visto resultados promisorios para estas herramientas, recomendamos a los negocios precaución al considerarlas como la suma total de su camino en el aprendizaje de máquinas. Tal y como se advirtió en H2O website, "todavía hay una buena cantidad de conocimiento y experiencia en ciencia de datos que es requerida para producir modelos de alto rendimiento de machine learning.". La confianza ciega en técnicas de automatización también incrementa el riesgo de introducir sesgos éticos o tomar decisiones que desfavorecen a minorías. Si bien éstas herramientas son un punto de partida para generar modelos entrenados útiles, promovemos la búsqueda de data scientists con experiencia para validar y refinar los resultados.

Certificación Binaria

PROBAR

Como el uso de contenedores, el despliegue de un gran número de servicios por parte de equipos autónomos y aumento en la velocidad de la entrega continua se han convertido en una práctica muy común en muchas organizaciones, ha surgido la necesidad de controlar la seguridad de software al momento de despliegue de forma automatizada. La certificación binaria (Binary attestation) es una técnica para el control de la seguridad al momento de despliegue; esto significa que nos permite verificar criptográficamente que una imagen binaria está autorizada para su despliegue. Usando esta técnica, un certificador, un proceso de compilación automatizado o un equipo de seguridad firma los archivos binarios autorizando que pueden ser puestos a producción siempre y cuando hayan pasado los controles de calidad y pruebas requeridas. Servicios como la

autorización binaria de GCP habilitada por Grafeas y herramientas como in-toto y Docker Notary permite la creación de certificaciones y validación de firmas de imágenes antes del despliegue.

Entrega continua para modelos de machine learning (CD4ML)

PROBAR

Con el aumento en popularidad de las aplicaciones basadas en machine learning, y la complejidad técnica que significa construirlas, nuestros equipos dependen mucho de la entrega continua para machine learning (CD4ML) para entregar estas aplicaciones de manera segura, rápida y sustentable. CD4ML es la disciplina de traer los principios y prácticas de entrega continua a las aplicaciones de machine learning. Elimina los largos ciclos entre el entrenamiento de los modelos y su despliegue a producción. CD4ML también elimina las transferencias manuales entre distintos equipos, ingenieras/os de datos, científicas/os de datos e ingenieras/os de machine learning en el proceso de extremo a extremo de construcción y despliegue de un modelo servido por una aplicación ML. Utilizando CD4ML nuestros equipos han logrado implementar de manera exitosa el versionamiento, prueba y despliegue automatizado de todos los componentes de sus aplicaciones basadas en ML: datos, modelos y código.

Descubrimiento de datos

PROBAR

Uno de los principales puntos de fricción para data scientists, en su flujo de trabajo, es ubicar los datos que necesitan, darles sentido y evaluar si es confiable usarlos. Esto sigue siendo un desafío debido a la falta de metadatos sobre las fuentes de datos disponibles y la falta de funcionalidad adecuada y necesaria para buscar y localizar datos. Alentamos a los equipos que están proporcionando conjuntos de datos analíticos o construyendo plataformas de datos para que el descubrimiento de datos sea una función de primera clase en sus entornos; para proporcionar la capacidad de localizar fácilmente los datos disponibles, detectar su calidad, comprender su estructura, linaje y tener acceso a ellos. Tradicionalmente, esta función ha sido proporcionada por soluciones de catalogación de datos inflados. En los últimos años, hemos visto el crecimiento de proyectos de código abierto que están mejorando las experiencias de las/os desarrolladoras/es, tanto para proveedores de datos como para consumidores de datos, para hacer una cosa realmente bien: hacer que los datos sean reconocibles. Amundsen de Lyft, y WhereHows de LinkedIn están entre estas herramientas. Lo que nos gusta ver es un cambio en el comportamiento de los proveedores para compartir intencionalmente los metadatos que ayudan a la capacidad de descubrimiento en favor de las herramientas de descubrimiento que infieren información de metadatos parcial de silos de bases en datos de aplicaciones.

Función de aptitud para montículos de dependencias

PROBAR

Muchos equipos y organizaciones no tienen métodos formales o consistentes de hacer seguimiento a las dependencias técnicas en su software. Este problema muchas veces aparece cuando el software tiene que ser modificado y el uso de una versión obsoleta de una biblioteca, API o componente, ocasiona trabas o demoras. La función de aptitud para montículos de dependencias es una técnica que introduce una función de aptitud de arquitectura evolutiva específica para hacer seguimiento a estas dependencias a lo largo del tiempo. De manera que se da un indicio del trabajo posiblemente requerido y si un problema potencial está mejorando o empeorando.

Design systems

PROBAR

A medida que el desarrollo de aplicaciones se vuelve cada vez más dinámico y complejo, es un reto conseguir la entrega eficiente de productos accesibles y usables que tengan estilos consistentes. Design systems define una colección de patrones de diseño, librerías de componentes y buenas prácticas de diseño y desarrollo que aseguran la consistencia en el desarrollo de productos digitales. Vemos a design systems como una herramienta adicional, útil cuando trabajamos con múltiples equipos y disciplinas en el desarrollo de producto, porque permiten a los equipos enfocarse en retos más estratégicos del producto sin necesidad de reinventar la rueda cada vez que se necesita agregar un componente visual. Los tipos de componentes y herramientas que uses para crear design systems pueden variar mucho.

Herramientas de seguimiento para experimentación con machine learning

PROBAR

El trabajo cotidiano de machine learning implica una serie de experimentos al seleccionar el enfoque del modelado, la topología de red, datos de entrenamiento y varias optimizaciones y ajustes al modelo. Ya que varios de estos modelos aún son difíciles de interpretar o explicar, los científicos de datos deben usar su experiencia e intuición para hipotetizar cambios y medir el impacto de los mismos en el rendimiento general del modelo. Dado que estos modelos se han vuelto cada vez más comunes en sistemas de negocio, varias herramientas de seguimiento para experimentación con machine learning han surgido para ayudar a los investigadores a llevar registro de estos experimentos y su trabajo metódicamente. Aunque no hay

TÉCNICAS

Las redes neuronales profundas han sido tomadas en cuenta por su precisión en una amplia variedad de problemas.

(La explicabilidad como criterio-relevante al seleccionar un modelo)

TÉCNICAS

El complejo panorama tecnológico de hoy exige tratar las políticas de seguridad como código: definir y mantener esas políticas bajo control de versionamiento, validarlas automáticamente, desplegarlas automáticamente y monitorear su desempeño.

(Política de seguridad como código)

Zhong Tai es un enfoque para entregar modelos de negocio encapsulados. Está diseñado para ayudar a una nueva generación de pequeños negocios que entregan servicios de primera clase sin los costes de infraestructura de las empresas tradicionales

(Zhong Tai)

una ganadora, herramientas como MLflow o Weights & Biases y plataformas como Comet o Neptune han introducido rigor y reproductibilidad en todo el flujo de trabajo de machine learning. También facilitan la colaboración y ayudan a convertir la ciencia de datos de un esfuerzo solitario a un esfuerzo de equipo.

La explicabilidad como criterio-relevante al seleccionar un modelo

PROBAR

Las redes neuronales profundas han sido tomadas en cuenta por su precisión en una amplia variedad de problemas. Alcanzados los suficientes datos de capacitación y la selección de una apropiada topología, estos modelos alcanzan y exceden las capacidades humanas en selectos campos de problemas. Sin embargo, resultan ser por naturaleza opacos. Aunque algunas partes de los modelos se pueden reutilizar mediante el aprendizaje por transferencia, pocas veces somos capaces de atribuir algún significado comprensible para los humanos a estos elementos. Por el contrario, un modelo explicativo es aquel que nos permite entender cómo se tomó cierta decisión. Por ejemplo, un árbol de decisión produce una cadena de inferencia que describe el proceso de clasificación. La explicabilidad se vuelve crítica en ciertas industrias reguladas o cuando nos preocupa el impacto ético de una decisión. A medida que estos modelos se adoptan más ampliamente en sistemas críticos del negocio es importante tomar en cuenta la explicabilidad como un criterio relevante al seleccionar un modelo. A pesar de su poder, las redes neuronales podrían no ser una elección acertada cuando existen estrictos requerimientos de explicabilidad.

Política de seguridad como código

PROBAR

Las políticas de seguridad son reglas y procedimientos que protegen a nuestros sistemas de amenazas y alteraciones. Por ejemplo, las políticas de control de acceso definen y resguardan quiénes pueden acceder a qué tipo de servicios y de recursos y bajo qué circunstancias; o las políticas de seguridad de redes pueden limitar dinámicamente la velocidad del tráfico a un servicio en particular. El complejo panorama tecnológico de hoy exige tratar las políticas de seguridad como código: definir y mantener esas políticas bajo control de versionamiento, validarlas automáticamente, desplegarlas automáticamente y monitorear su desempeño. Herramientas tales como el Open Policy Agent o plataformas como Istio proveen mecanismos flexibles de definición y ejecución de políticas que apoyan la práctica de tratar las políticas de seguridad como código.

Sidecars para seguridad de endpoints

PROBAR

Muchas de las soluciones técnicas que creamos hoy en día se ejecutan en entornos polycloud o nube híbrida cada vez más complejos con múltiples componentes y servicios distribuidos. En tales circunstancias, aplicamos dos principios de seguridad al inicio de la implementación: red Zero trust donde se recomienda nunca confiar en la red y siempre hacer verificaciones, y el principio del privilegio mínimo, otorgando los permisos mínimos necesarios para realizar un trabajo en particular. Los Sidecars para seguridad de endpoints son una técnica común que utilizamos para implementar estos principios y así cumplir los controles de seguridad en cada endpoint del componente. Ej: APIs de servicios, almacenes de datos, control de interfaces de Kubernetes. Hacemos esto usando un sidecar fuera de proceso: un proceso o un contenedor que se implementa y programa con cada servicio que comparte el mismo contexto de ejecución, host e identidad. Open Policy Agent y Envoy, son herramientas que implementan esta técnica. Los Sidecards para seguridad de endpoints minimizan la huella confiable en un endpoint local en lugar del perímetro de la red. Nos gusta ver que la responsabilidad de la configuración de la política de seguridad del sidecar recae en el equipo responsable del endpoint y no en un equipo centralizado separado.

Zhong Tai

PROBAR

Zhong Tai se ha convertido en una palabra de moda en la industria de TI China por años, pero todavía tiene que popularizarse en el oeste. En su núcleo, Zhong Tai es un enfoque para entregar modelos de negocio encapsulados. Está diseñado para ayudar a una nueva generación de pequeños negocios que entregan servicios de primera clase sin los costes de infraestructura de las empresas tradicionales y habilitando a las organizaciones a proporcionar servicios innovadores a velocidades vertiginosas. La estrategia de Zhong Tai fue propuesta originalmente por Alibaba y pronto fue seguida por muchas compañías de internet Chinas, porque su modelo de negocio es nativo digital para replicar los nuevos mercados y sectores. Hoy en día, más firmas Chinas están utilizando Zhong Tai como palanca para la transformación digital.

BERT

EVALUAR

BERT (Bidirectional Encoder Representations from Transformers) es un nuevo método de representación de lenguaje pre-entrenado publicado por investigadores de Google en Octubre de 2018. BERT ha modificado significativamente el ecosistema del procesamiento del lenguaje natural (PLN)

obteniendo resultados vanguardistas en una amplia gama de tareas de PLN. Basado en una arquitectura Transformer, durante el entrenamiento aprende del contexto de un token tanto por la derecha como por la izquierda. Google también ha publicado modelos pre-entrenados de propósito general para BERT que han sido entrenados en un gran corpus de texto no etiquetado, incluyendo la Wikipedia. Developers pueden usar y ajustar estos modelos pre-entrenados para los datos específicos de su tarea y conseguir grandes resultados. Hablamos acerca de transferencia de aprendizaje en PLN en nuestra edición de Abril de 2019 del Radar; BERT y sus sucesores continúan haciendo que la transferencia de aprendizaje para PLN sea un campo muy interesante, reduciendo significativamente el esfuerzo para usuarios que lidian con la clasificación de texto.

Malla de datos

EVALUAR

Malla de datos es un paradigma de arquitectura que desbloquea datos analíticos a escala; rápidamente desbloquea accesos a un número cada vez mayor de conjuntos distribuidos de datos de dominio, para una proliferación de escenarios de consumo tales como aplicaciones de aprendizaje automático, análisis o uso intensivo de datos en toda la organización. Malla de datos aborda los modos de fallas comunes de los data lakes centralizados tradicionales o de la arquitectura de plataforma de datos, con un cambio desde el paradigma centralizado de un lake, o su predecesor, el data warehouse. Malla de datos cambia a un paradigma que traza desde una arquitectura distribuida moderna: considerando dominios como los asuntos de primera clase, aplicando platform thinking para crear una infraestructura de datos de autoservicio, tratamiento de datos como un producto, e implementando estandarización abierta para habilitar un ecosistema de productos de datos distribuidos inter-operables.

Pruebas de sesgo ético

EVALUAR

Durante el año pasado, hemos visto cambios en el interés en torno al aprendizaje automático y redes neuronales profundas en particular. Hasta ahora, el desarrollo de herramientas y técnicas ha sido impulsado por la emoción de las capacidades singulares de estos modelos. Sin embargo, actualmente existe una preocupación creciente de que estos modelos pueden causar daño involuntario. Por ejemplo, un modelo puede ser entrenado para hacer decisiones de crédito rentable al excluir simplemente a solicitantes desfavorecidos. Afortunadamente, estamos viendo un interés creciente en pruebas de sesgo ético que ayudará a descubrir decisiones potencialmente dañinas. Herramientas como lime, AI Fairness 360 o What-if pueden ayudar a descubrir imprecisiones que resultan de grupos subrepresentados en datos de entrenamiento y herramientas de visualización tales como Google Facets o Facets Dive pueden ser usados para descubrir subgrupos dentro de un corpus de datos de entrenamiento. No obstante, este es un campo en desarrollo y esperamos que normas y prácticas específicas sobre pruebas de sesgo ético surjan con el tiempo.

Aprendizaje Federado

EVALUAR

El entrenamiento de los modelos generalmente requiere recolectar los datos desde su fuente y transportarlos a una localización centralizada donde corre el algoritmo de entrenamiento.Esto se vuelve particularmente problemático cuando los datos utilizados para el entrenamiento consisten en información personal identificable. Nos incentiva el auge del aprendizaje federado como un método de entrenamiento para diversos sets de datos que permite preservar la privacidad. Las técnicas utilizadas en el aprendizaje federado permiten que los datos permanezcan en el dispositivo del usuario y bajo su control, contribuyendo igualmente con una colección de datos para el entrenamiento de modelos. Para eso, el dispositivo actualiza un modelo independientemente; luego los parámetros del modelo, en lugar de los datos, son combinados en una vista centralizada.
El ancho de banda y las limitaciones computacionales del dispositivo representan retos técnicos significativos, pero nos gusta como el aprendizaje federado deja al usuario en control de su propia información personal.

JAMstack

EVALUAR

La tendencia que empezó como backend como servicio para aplicaciones móvil nativas hace ya varios años, ahora se está popularizando con aplicaciones web. Estamos viendo frameworks de trabajo como Gatsby.js que combinan la generación de sitios estáticos y el renderizado del lado de cliente con APIs de terceros. Referido como JAMstack (donde JAM significa JavaScript, API, y Markup), este enfoque puede proveer una valiosa experiencia de usuario en aplicaciones web que dependen mayormente de APIs y ofertas de SaaS. Ya que el HTML se renderiza o bien en el navegador o durante el tiempo de compilación, el modelo de despliegue es el mismo que usan los sitios generados de manera completamente estática, con todos sus beneficios: la superficie de ataque en el servidor es pequeña y se puede lograr gran rendimiento con bajo uso de recursos. Despliegues como estos también son ideales para una red de distribución de contenidos. De hecho, jugamos con la idea de nombrar a esta técnica aplicaciones CDN first (CDN primero).

TÉCNICAS

Nos incentiva el auge del aprendizaje federado como un método de entrenamiento para diversos sets de datos que permite preservar la privacidad.

(Aprendizaje Federado)

JAMstack puede proveer una valiosa experiencia de usuario en aplicaciones web que dependen mayormente de APIs y ofertas de SaaS.

(JAMstack)

TÉCNICAS

En nuestra experiencia, grandes ingenieras/os no se guían por el output individual, sino por trabajar en equipos fantásticos.

(10x engineers)

Vincular registros conservando la privacidad (PPRL) con el filtro de Bloom

EVALUAR

Cuando existe una clave compartida, vincular registros a partir de varios proveedores de datos resulta una tarea sencilla. Sin embargo, es posible que no siempre exista una; y aún cuando la haya, puede que no sea buena idea mostrarla por cuestiones de privacidad. La técnica de vincular registros conservando la privacidad (PPRL) con el filtro de Bloom permite el vínculo probabilístico de registros de varios proveedores de datos, sin revelar información de carácter personal. Por ejemplo, para vincular datos de dos proveedores, cada proveedor cifra estos datos personales mediante el filtro de Bloom para obtener claves vinculadas criptográficamente que luego se envían a través de un canal seguro. Tras recibir esos datos, los registros pueden vincularse calculando el índice de similitud entre conjuntos de estas claves de cada proveedor. Hemos observado que, entre otras técnicas, la de PPRL con filtros de Bloom sería escalable para conjuntos de datos de gran volumen.

Ciclos de aprendizaje semi-supervisados

EVALUAR

Los ciclos de aprendizaje semi-supervisados son una clase de flujo de trabajo de aprendizaje automático iterativo que aprovecha las relaciones que pueden encontrarse en datos no etiquetados. Estas técnicas pueden mejorar modelos al combinar grupos de datos etiquetados y no etiquetados de varias maneras. En otros casos, comparan modelos entrenados con diferentes subcolecciones de los datos. A diferencia del aprendizaje no supervisado, donde la máquina infiere clases de datos no etiquetados o de técnicas supervisadas o la colección de entrenamiento está completamente etiquetada, las técnicas semi-supervisadas sacan partido de una pequeña colección de datos etiquetados y una colección mucho más grande de datos no etiquetados.El aprendizaje semi-supervisado también está estrechamente relacionado a técnicas de aprendizaje activas donde a un humano se le dirige para que etiquete selectivamente puntos de datos ambiguous. Ya que los expertos humanos capaces de etiquetar datos de manera certera son un recurso escaso y que el etiquetar es a menudo la actividad que consume más tiempo en el flujo de trabajo del aprendizaje automático, las técnicas semi-supervisadas reducen los costes y hacen al aprendizaje automático más accesible a una nueva clase de usuarios.

10x engineers

RESISTIR

El antiguo término 10x engineer ha sido sometido a escrutinio estos pasados meses. Un hilo ampliamente distribuido en Twitter sugiere esencialmente que las compañías deben excusar comportamientos antisociales y dañinos para retener ingenieras/os que son percibidas/os como altamente eficientes (a nivel de output individual). Afortunadamente, muchas personas en la red social realizaron bromas sobre el concepto, pero el estereotipo del rol de “desarrollador/a rockstar” es todavía generalizado. En nuestra experiencia, grandes ingenieras/os no se guían por el output individual, sino por trabajar en equipos fantásticos. Es más eficiente construir equipos de gente con talento, tanto experiencias como backgrounds variados, que proporcionen los ingredientes precisos para el trabajo en equipo, el aprendizaje y la mejora continua. Estos equipos 10x pueden moverse más rápido, escalar rápidamente y ser mucho más resilientes, sin necesidad de justificar malos comportamientos.

Integración de front-end a través de artefactos

RESISTIR

Cuando los equipos adoptan el concepto de micro frontends disponen de un número de patrones para integrar los micro frontends individuales en una aplicación. Como siempre, también existen anti-patrones. Uno común en este caso, es la integración de front-end a través de artefacto. Para cada micro frontend se construye un artefacto, normalmente un paquete NPM, que se sube a un repositorio. Un paso posterior, a veces en un flujo de construcción distinto, combina los paquetes individuales en un paquete final que contiene todos los micro frontends. Desde una perspectiva puramente técnica esta integración en tiempo de construcción da lugar a una aplicación funcional. Sin embargo, la integración a través de artefacto implica que, por cada cambio, el artefacto completo tiene que volverse a construir, lo que lleva tiempo y, muy probablemente, impacte negativamente en la experiencia de la persona desarrolladora. Lo que es peor, esta forma de integrar frontends también introduce dependencias directas entre los micro frontends en tiempo de construcción y, por tanto, provoca un mayor sobrecoste de coordinación.

TÉCNICAS

Las arquitecturas Lambda pinball característicamente sueles perder de vista la lógica de dominio importante en tanto enredo de Lambdas, buckets y colas, a medida que las solicitudes de estas rebotan, se incrementa entonces la complejidad de los gráficos en los servicios en la nube

(Lambda pinball)

Lambda pinball

RESISTIR

Llevamos ya un par de años construyendo arquitecturas serverless en nuestros proyectos y hemos notado que es muy fácil caer en la trampa de construir un monolito distribuido. Las arquitecturas Lambda pinball característicamente suelen perder de vista la lógica importante de dominio entre tanto enredo de Lambdas, buckets y colas, a medida que las solicitudes de estas rebotan, se incrementa entonces la complejidad de los gráficos en los servicios en la nube. Usualmente, es complicado testearlas como unidades, y las necesidades de la aplicación deben ser testeadas como un todo integrado. Un patrón que podemos utilizar para prevenir estas arquitecturas de pinball es diferenciar entre interfaces públicas y publicadas y aplicar límites de dominio con interfaces publicadas entre ellas.

Paridad de funcionalidades en migraciones heredadas

RESISTIR

Hemos descubierto que cada vez más y más organizaciones tienen la necesidad de reemplazar sistemas heredados para mantenerse al día con las demandas de sus clientes (tanto internos como externos). Un antipatrón que continuamos viendo es la paridad de funcionalidades en migraciones legadas, el deseo de retener una paridad de funcionalidad con lo antiguo. Consideramos que esto es una enorme oportunidad perdida. A menudo, los sistemas antiguos se han sobrecargado con el paso del tiempo, con muchas funcionalidades que los usuarios no utilizan (50% de acuerdo al reporte de 2014 de Standish Group) y procesos de negocio que han evolucionado con el tiempo. Reemplazar estas funcionalidades es perder el tiempo. Nuestro consejo: Convence a tus clientes a dar un paso atrás y entender lo que sus usuarios necesitan actualmente, para luego priorizar estas necesidades en base a resultados comerciales y métricas — lo que usualmente es más fácil de decir que de hacer. Esto significa que se debe conducir una investigación de usuarios y aplicar prácticas modernas de desarrollo de producto en lugar de simplemente reemplazar las existentes.

PLATAFORMAS

Apache Flink
PROBAR

Apache Flink ha visto una creciente adopción desde nuestra evaluación inicial en 2016. Flink es reconocido como el motor líder de procesamiento de flujos y también ha madurado gradualmente en los campos del procesamiento por lotes (batch) y del machine learning. Uno de los diferenciadores clave de Flink con respecto a otros motores de procesamiento de flujos es el uso de puntos de control consistentes del estado de una aplicación. En caso de fallo, la aplicación se reinicia y su estado se carga desde el último punto de control, de modo que la aplicación puede seguir procesando como si el fallo nunca hubiera ocurrido. Esto nos ayuda a reducir la complejidad de la construcción y operación de sistemas externos para la tolerancia a fallos. Vemos cada vez más empresas que utilizan Flink para construir su plataforma de procesamiento de datos.

Apollo Auto
PROBAR

Durante mucho tiempo exclusiva para los gigantes tecnológicos, la tecnología de vehículos autónomos ya está al alcance de todo el mundo, como ha demostrado Apollo Auto. El objetivo del programa Apolo de Baidu es convertirse en el Android de la industria del vehículo autónomo. La plataforma Apollo posee componentes como percepción, simulación, planeamiento y control inteligente que permiten a las compañías automotoras integrar sus propios sistemas de conducción autónoma en el hardware de sus vehículos.

La comunidad de personas desarrolladoras es aún nueva pero con una gran cantidad de proveedores incorporándose para contribuir con más ports. Uno de nuestros proyectos fue ayudar a nuestro cliente a completar exámenes de permiso de conducir para vehículos autónomos con el sistema de piloto automático basado en Apollo. Apollo también provee un enfoque de arquitectura evolutiva para adoptar funcionalidades avanzadas de manera gradual, lo que nos permite integrar más sensores y funciones de un modo ágil e iterativo.

GCP Pub/Sub
PROBAR

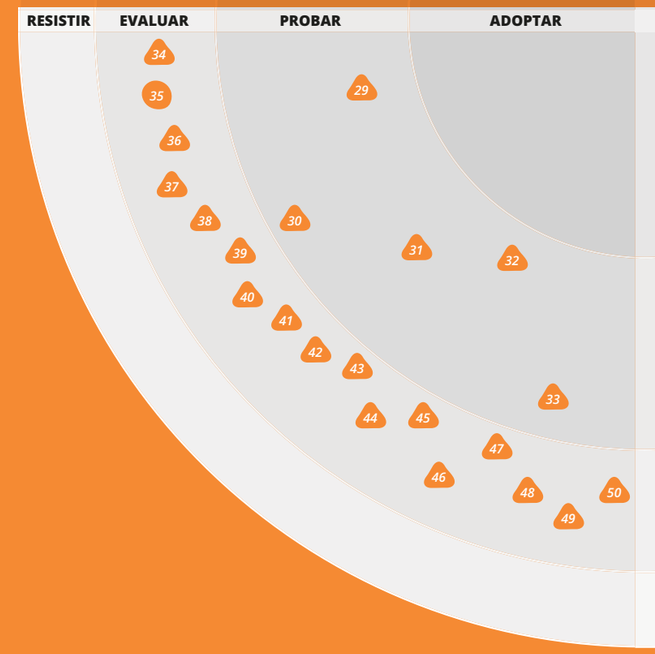
GCP Pub/Sub es la plataforma de transmisión de eventos de Google Cloud. Es una pieza de infraestructura popular para muchas de nuestras arquitecturas que ejecutan Google Cloud Platform, incluída la ingestión masiva de eventos, la comunicación de cargas de trabajo sin servidor y los flujos de trabajo de procesamiento de datos de transmisión. Una de sus características únicas es el soporte de suscripciones pull y push: suscribirse para recibir todos los mensajes publicados disponibles en el momento de la suscripción, o enviar mensajes a un endpoint en particular. Nuestros equipos han disfrutado de su confiabilidad y escala, y de que funciona tal como se anuncia.

ADOPTAR

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

EVALUAR
34. AWS Cloud Development Kit
35. Azure DevOps
36. Azure Pipelines
37. Crowdin
38. Crux
39. Delta Lake
40. Fission
41. FoundationDB
42. GraalVM
43. Hydra
44. Kuma
45. MicroK8s
46. Oculus Quest
47. ONNX
48. Contenedores sin raíces
49. Snowflake
50. Teleport

RESISTIR



PLATAFORMAS

Apache Flink es reconocido como el motor líder de procesamiento de flujos y también ha madurado gradualmente en los campos del procesamiento por lotes (batch) y del machine learning.

(Apache Flink)

El objetivo del programa Apolo de Baidu es convertirse en el Android de la industria del vehículo autónomo.

(Apollo Auto)

Mongoose OS
PROBAR

Mongoose OS continúa siendo uno de nuestros sistemas operativos de microcontroladores de código abierto y marcos de desarrollo de firmware integrados preferidos. Vale la pena señalar que Mongoose OS llena un vacío notable para las personas desarrolladoras de software integrado: el vacío entre el firmware Arduino adecuado para la creación de prototipos y los SDK nativos de microcontroladores bare metal. Nuestros equipos han utilizado con éxito la nueva plataforma de administración de dispositivos de extremo a extremo de Cesanta, mDash, para proyectos de hardware greenfield a pequeña escala. Los principales proveedores de la plataforma en la nube de Internet de las cosas (IoT) admiten hoy el marco de desarrollo de Mongoose OS para la gestión de sus dispositivos, conectividad y actualizaciones de firmware por aire (OTA). Desde la última vez que informamos sobre Mongoose OS, la cantidad de placas y microcontroladores compatibles ha crecido hasta incluir STM, Texas Instruments y Espressif. Continuamos disfrutando de su compatibilidad sin problemas para las actualizaciones de OTA y su seguridad integrada a nivel de dispositivo individual.

ROS
PROBAR

ROS (Robot Operating System) es un conjunto de librerías y herramientas para ayudar a las personas desarrolladoras a crear aplicaciones de robot. Es un framework que provee abstracciones sobre el hardware, drivers de dispositivos, librerías, visualizadores, transmisión de mensajes, administrador de paquetes, y mucho más. Apollo Auto está basado en ROS. En otros de nuestros proyectos de simulación ADAS, hemos utilizado también el sistema de mensajes (bag) de ROS. La tecnología no es nueva, pero ha vuelto a captar la atención de los/las desarrolladores/as con el desarrollo de ADAS.

AWS Cloud Development Kit
EVALUAR

Para muchos de nuestros equipos, Terraform se ha convertido en la opción predeterminada para definir la infraestructura de la nube. Sin embargo, algunos de nuestros equipos han estado experimentando con AWS Cloud Development Kit (AWS CDK) y les ha gustado lo que han visto hasta ahora. En particular, les gusta el uso de lenguajes de programación de primera clase en lugar de archivos de configuración que les permite usar herramientas existentes, enfoques de prueba y habilidades. Al igual que otras herramientas similares, aún se necesita ser cautelosos para garantizar que las implementaciones sigan siendo fáciles de entender y mantener. Dado que el soporte para C# y Java llegará pronto e ignorando, por ahora, algunos vacíos en la funcionalidad, creemos que vale la pena ver AWS CDK como alternativa a otros enfoques basados en archivos de configuración.

Azure DevOps
EVALUAR

Los servicios de Azure DevOps incluyen un conjunto de servicios administrados, como repositorios Git alojados, pipelines de CI/CD, herramientas de prueba automatizadas, herramientas de administración de trabajos pendientes y repositorio de artefactos. Los pipelines de Azure DevOps han ido madurando con el tiempo. Nos gusta especialmente su capacidad para definir Pipelines como código y su ecosistema de extensiones en el mercado de Azure DevOps. Al momento de escribir esto, nuestros equipos aún se encuentran con algunas características inmaduras, incluída la falta de una interfaz de usuario efectiva para la visualización y navegación de pipelines, y la incapacidad de activar un pipeline a partir de artefactos u otras pipelines.

Azure Pipelines
EVALUAR

Azure Pipelines es un producto de Azure DevOps que ofrece soluciones cloud para implementar pipelines como código para proyectos alojados en el servidor Azure DevOps Git u otra solución Git como GitHub o Bitbucket. La parte interesante de esta solución es la habilidad de ejecutar scripts en agentes Linux, MacOS y Windows sin el sobrecoste de tener que gestionar una máquina virtual. Representa un gran paso adelante, especialmente para equipos que trabajan en entornos Windows con soluciones en el framework .NET; también estamos evaluando este servicio para entrega continua en iOS.

Crowdin
EVALUAR

La mayoría de los proyectos que tienen soporte en varios idiomas, comienzan con el equipo desarrollando funcionalidades en un idioma, para luego ocuparse del resto a través de traducciones mediante correos electrónicos y hojas de cálculo. Aunque este simple proceso funciona, puede irse rápidamente de las manos. Probablemente tengas que responder la misma pregunta una y otra vez a los diferentes traductores de los distintos idiomas, quitando la energía a la colaboración entre traductores, correctores de las traducciones, y el equipo de desarrollo. Crowdin es una de las tantas plataformas que ayudan a facilitar el proceso de localización de idiomas de tus proyectos. Con Crowdin, el equipo de desarrollo puede continuar desarrollando funcionalidades y la plataforma simplifica el proceso de los textos que deben ser traducidos, mediante un flujo de trabajo en línea. Nos gusta que Crowdin empuje a los equipos a incorporar continua e incrementalmente las traducciones, en vez de tener que manejarlas todas juntas al final.

Crux
EVALUAR

Crux es una base de datos de documentos de código libre con consultas gráficas bitemporales. La mayoría de los sistemas de bases de datos son temporales, lo que significa que nos ayudan a modelar los hechos junto al momento en que ocurrieron. Los sistemas de bases de datos bitemporales te permiten modelar no solo el momento válido en que ocurrió el hecho, sino también el tiempo de transacción en que fue recibido. Si necesitas un almacén de documentos con capacidades gráficas para consultar el contenido, prueba Crux. Actualmente está en alfa y carece de soporte SQL, pero se puede utilizar una interfaz de consulta de Datalog para leer y atravesar las relaciones.

Delta Lake
EVALUAR

Delta Lake es una capa de almacenamiento de open-source de Databricks que intenta llevar las transacciones al procesamiento de big data. Uno de los problemas que a menudo encontramos al usar Apache Spark es la falta de transacciones ACID. Delta Lake se integra con la API de Spark y resuelve este problema mediante el uso de un registro de transacciones y archivos de Parquet versionados. Su aislamiento serializable, permite que lectores y escritores puedan trabajar sobre archivos Parquet simultáneamente. Dentro de las bien recibidas características se incluye la aplicación de esquemas al escribir y versionar, lo que nos permite consultar y volver a versiones anteriores de datos de ser necesario. Hemos comenzando a usarlo en algunos de nuestros proyectos y nos resulta interesante.

Fission
EVALUAR

El ecosistema serverless de Kubernetes está creciendo. En radares anteriores hemos hablado acerca de Knative; ahora estamos viendo a Fission ganar terreno. Fission permite las/os desarrolladoras/es enfocarse en escribir funciones de vida corta y transformarlas a peticiones HTTP, mientras que el framework maneja el resto de las conexiones y automatización de recursos Kubernetes por detrás escenas. Fission también permite crear funciones compuestas, se integra con proveedores externos por medio de web hooks y permite automatizar la administración de la infraestructura de Kubernetes.

FoundationDB
EVALUAR

FoundationDB es una base de datos multimodelo open-source que fue comprada por Apple en 2015 y cuyo código se liberó en 2018. El núcleo de FoundationDB es un almacenamiento clave-valor que provee transacciones serializables estrictas. Uno de los aspectos más interesantes de FoundationDB es su concepto de capas para ofrecer modelos adicionales. Estas capas son esencialmente componentes sin estado construidos sobre dicho almacenamiento clave-valor, como por ejemplo Record layer y Document layer. FoundationDB pone el listón alto con su Simulation testing: donde corren diariamente pruebas donde se simula diferentes fallos de sistema. Su performance, testing riguroso y facilidad de manejo, hacen que FoundationDB no sólo esté dirigida para personas que buscan una base de datos sino también para aquellas que buscan construir sistemas distribuidos donde FoundationDB puede ser usada como núcleo primitivo sobre el que se construyen dichos sistemas.

GraalVM
EVALUAR

GraalVM es una máquina virtual universal de Oracle para correr aplicaciones escritas en lenguajes JVM, JavaScript, Python, Ruby, y R, así como C/C+ y otros lenguajes basados en LLVM. En su forma más simple, GraalVM puede ser usado como una MV con mayor rendimiento para JVM y otros lenguajes no-JVM soportados. Pero también nos permite escribir aplicaciones políglotas con muy poco impacto en el rendimiento; y su utilidad de Imagen Nativa (de momento solo disponible como una tecnología de Early Adopter) nos permite realizar una compilación anticipada de código Java a ejecutables autónomos para

PLATAFORMAS

GraalVM ha despertado mucho entusiasmo en la comunidad de Java, y varios frameworks de Java (incluyendo a Micronaut, Quarkus, y Helidon) ya están sacando provecho suyo.

(GraalVM)

PLATAFORMAS

la interoperabilidad entre herramientas ha supuesto todo un reto. ONNX puede ayudar.

(ONNX)

Kuma es una malla de servicio agnóstica a nivel de plataforma para Kubernetes, máquinas virtuales y entornos físicos.

(Kuma)

un inicio más rápido y con un menor uso de memoria. GraalVM ha despertado mucho entusiasmo en la comunidad de Java, y varios frameworks de Java (incluyendo a Micronaut, Quarkus, y Helidon) ya están sacando provecho suyo.

Hydra

EVALUAR

No todos necesitamos una solución OAuth2 con alojamiento propio pero, en caso de necesitarla, consideramos que Hydra —un proveedor de conexiones OpenID y servidor OAuth2 de open-source que cumple con estas especificaciones— es bastante útil. Nos gusta mucho que Hydra no provea ninguna solución de gestión de identidades propia, ya que esto nos permite integrar cualquier tipo de gestión de identidades que ya tengamos con Hydra mediante una API limpia. Esta completa separación de la identidad del resto de OAuth2 facilita la integración de Hydra con un entorno de autenticación ya existente.

Kuma

EVALUAR

Kuma es una malla de servicio agnóstica a nivel de plataforma para Kubernetes, máquinas virtuales y entornos físicos. Kuma está implementada como un plano de control por encima de Envoy y como tal, puede instrumentalizar cualquier tráfico de las capas 4 y 7 para asegurar, observar, enrutar y mejorar la conectividad entre servicios. La mayor parte de las implementaciones de mallas de servicios están enfocadas de manera nativa al ecosistema de Kubernetes, lo que en sí mismo no está mal pero obstaculiza la adopción de mallas de servicio para aplicaciones existentes fuera de Kubernetes. En vez de esperar a que termine un gran esfuerzo de transformación de plataforma, ahora puedes usar Kuma y modernizar tu infraestructura de red.

MicroK8s

EVALUAR

Ya hemos hablado de Kubernetes anteriormente y todavía sigue siendo la primera elección para desplegar y gestionar contenedores en clústers de producción. Sin embargo, cada vez es más difícil proveer la misma experiencia para las personas que desarrollan de forma desconectada. Entre otras opciones hemos encontrado bastante útil a MicroK8s. Para instalar el snap de MicroK8, elige el canal de lanzamiento (stable, candidate, beta o edge) y luego, con algunos comandos, puedes tener a Kubernetes en ejecución. También es posible monitorear los próximos lanzamientos y elegir actualizar tu instalación automáticamente.

Oculus Quest

EVALUAR

Hemos seguido la trayectoria de AR/VR (realidad aumentada/virtual) por largo tiempo en nuestro Radar, pero su atractivo se ha limitado a plataformas y opciones de enlace específicos. Oculus Quest ha revolucionado el mercado al convertirse en uno de los primeros visores de VR autónomos para el público general que no requiere de ningún tipo de enlace o soporte aparte de un teléfono inteligente. Este dispositivo abre el camino para dar una mayor visibilidad a las aplicaciones de VR, cuya demanda empujará el mercado hacia innovaciones más ambiciosas. Por ello, aplaudimos la democratización de VR que este dispositivo promueve y esperamos con ansias ver lo que depara el futuro.

ONNX

EVALUAR

El entorno de herramientas y marcos de trabajo para redes neuronales ha evolucionado a gran velocidad, pero la interoperabilidad entre ellas ha supuesto todo un reto. En la industria del aprendizaje de máquina (Machine Learning) es habitual diseñar un modelo y entrenarlo rápidamente con una herramienta, y después desplegarlo en otra para sacar conclusiones. Dado que los formatos internos de estas herramientas son incompatibles, necesitamos implementar y mantener conversores complejos para lograr esa compatibilidad. El formato Open Neural Network Exchange (ONNX) resuelve este problema. En ONNX, las redes neuronales se representan con gráficos usando especificaciones de operadores convencionales que, junto con un formato de serialización para los pesos entrenados, permite que los modelos de redes neuronales puedan transferirse de una herramienta a otra. Esto allana el camino para muchas posibilidades, como Model Zoo, una colección de modelos pre-entrenados en formato ONNX.

PLATAFORMAS

Teleport es una pasarela de seguridad para el acceso a infraestructuras nativas en la nube.

(Teleport)

Contenedores sin raíces
EVALUAR

En teoría, los contenedores se deben controlar desde el respectivo tiempo de ejecución del contenedor, sin privilegios de usuario root. Esto no es sencillo ni trivial pero cuando se consigue, reduce la superficie de ataque y previene muchos problemas de seguridad, en particular la escalación de privilegios fuera del contenedor. Ya desde hace tiempo se habla de los contenedores sin raíces, y es parte de la especificación del tiempo de ejecución de contenedores abiertos y sus estándares de implementación runc, parte de Kubernetes. Por otra parte, Docker 19.03 introduce contenedores sin privilegios, como una funcionalidad experimental. Aunque esta funcionalidad no va bien todavía con varias de otras funcionalidades como los controles de recursos cgroups y los perfiles de seguridad AppArmor.

Snowflake
EVALUAR

A menudo relacionamos el almacenamiento de datos (o “Data Warehousing”) a una infraestructura central que es difícil de escalar y administrar con las crecientes demandas alrededor de los datos. Snowflake, sin embargo, es una nueva solución de SQL Data Warehouse como servicio (SQL as a Service) construida desde cero para la nube. Con varias características cuidadosamente construidas tales como atomicidad a nivel de base de datos, soporte para datos estructurados y semiestructurados, funciones analíticas nativas, y sobre todo una separación clara de capas de almacenamiento, cómputo, y servicios, Snowflake aborda la mayoría de los desafíos a encontrarse en el área de Data Warehousing.

Teleport
EVALUAR

Teleport es una pasarela de seguridad para el acceso a infraestructuras nativas en la nube. Una de las características interesantes de Teleport es su capacidad para actuar como Autoridad de Certificación (CA) para tu infraestructura. Puedes emitir certificados de corta duración y construir accesos basados en roles (RBAC) más ricos para la infraestructura sobre Kubernetes (o solo para SSH). Al aumentar el foco en la seguridad de la infraestructura es importante mantener la pista de los cambios. Sin embargo, no todos los eventos requieren el mismo nivel de auditoría. Con Teleport puedes continuar registrando en el log la mayoría de los eventos, y a la vez proporcionar un plus al registrar también la actividad en la pantalla de usuario para las sesiones de root que tienen mayores privilegios.

HERRAMIENTAS

Commitizen
ADOPTAR

Commitizen es una herramienta para facilitar la simplificación del proceso de commit cuando se usa GIT. La herramienta te pide los campos requeridos y en los formatos adecuados para tu mensaje de commit. Permite varias convenciones de formatos de registro y también permite la creación de formatos propios vía adaptador. Esta herramienta es simple y ahorra tiempo, evitando rechazos más tarde en el proceso de commit.

ESLint
ADOPTAR

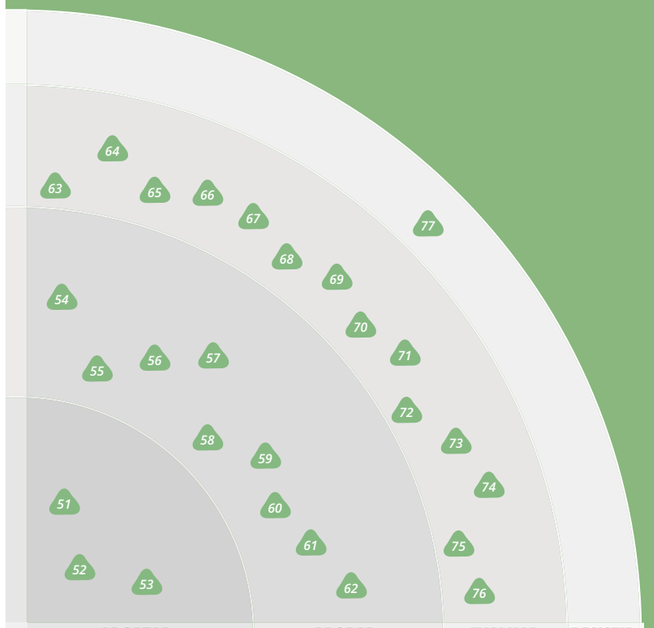
ESLint se está utilizando como estándar en muchos de nuestros proyectos. Como herramienta de linting para JavaScript tiene múltiples juegos de reglas, reglas recomendadas y extensiones para expandir a frameworks o JavaScript. Hemos visto cómo los equipos se apoyan fuertemente en ella para ayudarles a crear normas de código y forzar su cumplimiento, permitiendo así, el análisis en tiempo real del código durante el desarrollo. Puede utilizarse para estandarizar las prácticas de codificación obligando al cumpliento de las mejores prácticas y estilo de código, e identificar vulnerabilidades. Lo hace integrándose bien con la mayoría de los IDEs y proporcionando feedback en vivo mientras se codifica.
Sus reglas de estilo en particular arreglan automáticamente los errores de linting, haciendo que el proceso sea fluido y efectivo sin incurrir en un coste de desarrollo adicional. Las/os desarrolladoras/es pueden rápidamente alcanzar velocidad de crucero con las reglas, gracias a la documentación de la comunidad, que hace un buen trabajo explicando los patrones de codificación. A medida que ESLint se ha vuelto más común y potente, ha ganado tracción en la industria y una prueba de ello lo ilustra el equipo de TypeScript haya realizado un movimiento para soportar y trabajar con ESLint en vez de invertir en TSLint.

React Styleguidist
ADOPTAR

React Styleguidist es un entorno de desarrollo para los componentes de React. Contiene un servidor de desarrollo con capacidades hot reloading y genera una guía de estilos de HTML para compartir con equipos. La guía de estilos muestra la versión en vivo de cada componente en un solo lugar, con documentación para su uso y lista de sus props. Ya hemos mencionado en el pasado React Styleguidist como una UI de entorno de desarrollo, y con el tiempo se ha convertido en la herramienta líder en este espacio.

Bitrise
PROBAR

Construir, probar y desplegar aplicaciones móviles implica pasos complejos, especialmente cuando consideramos un pipeline que vaya desde el repositorio de código fuente a las tiendas de aplicaciones. Todos estos pasos pueden ser automatizados con scripts y pipelines de construcción en herramientas genéricas de CI/CD. Aun así, nuestros equipos han encontrado que Bitrise, una herramienta de CD de dominio específico para



ADOPTAR

51. Commitizen
52. ESLint
53. React Styleguidist

PROBAR

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

EVALUAR

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

RESISTIR

77. Azure Data Factory para orquestación

HERRAMIENTAS

Figma tiene las mismas funcionalidades que los programas de diseño como Sketch e Invision, pero permitiendo la colaboración con otras personas al mismo tiempo

(Figma)

Si estás construyendo una aplicación Java y utilizas Docker, debes considerar utilizar Google´s Jib.

(Jib)

aplicaciones móviles, es útil cuando no hay necesidad de integrar con pipelines de construcción de sistemas back-end. Bitrise es fácil de configurar y proporciona un conjunto completo de pasos pre-construidos que cubre la mayor parte de las necesidades del desarrollo móvil.

Dependabot
PROBAR

Mantener al día las dependencias de software es una labor que toma tiempo, pero por razones de seguridad es importante responder a dichas actualizaciones de manera oportuna. Podemos usar herramientas para hacer que el proceso sea lo más llevadero y automatizado posible. En la práctica nuestros equipos han tenido buenas experiencias con Dependabot. Se integra con repositorios de GitHub y verifica las dependencias automáticamente para buscar nuevas versiones. De ser necesario, Dependabot abrirá una pull request con las dependencias actualizadas.

Detekt
PROBAR

Detekt es una herramienta de análisis de código estático para Kotlin. Provee análisis de code smell e informes de complejidad con conjuntos de reglas altamente configurables. Se puede ejecutar desde línea de comandos, usar plugins, vía Gradle, SonarQube e IntelliJ. Nuestros equipos consideran muy valioso el uso de Detekt para mantener código de alta calidad. Cuando el análisis y la generación de informes están integrados en la construcción del pipeline, es importante que estos informes se revisen de manera regular y que los equipos reserven tiempo para actuar en lo que se descubra.

Figma
PROBAR

Uno de los grandes puntos de dolor en interacción y diseño visual es la falta de herramientas creadas para la colaboración. Aquí es donde Figma entra en escena. Tiene las mismas funcionalidades que los programas de diseño como Sketch e Invision, pero permite la colaboración con otras personas al mismo tiempo y ayuda a que descubras nuevas ideas junto con capacidades de colaboración en tiempo real. Nuestros equipos encuentran Figma muy útil, especialmente para habilitar y facilitar el trabajo de diseño remoto y distribuido. Adicionalmente a su capacidad de colaboración, Figma también ofrece una API que ayuda a mejorar el proceso de DesignOps.

Jib
PROBAR

La construcción de aplicaciones contenerizadas puede requerir de configuraciones complejas en entornos de desarrollo y en construcción de agentes. Si estás construyendo una aplicación Java y utilizas Docker, debes considerar utilizar Google´s Jib. Jib es un plugin open-source soportado por Maven y Gradle. El plugin Jib utiliza información de la configuración de tu *build* para construir la aplicación directamente como una imagen Docker, sin requerir Dockerfile ni Docker demon. Jib optimiza en capas de imágenes, prometiendo acelerar las builds consecuentes.

Loki
PROBAR

Loki es una herramienta de regresión visual que funciona con Storybook, la cual mencionamos previamente en el contexto de los entornos de desarrollo de la interfaz de usuario. Con unas pocas líneas de configuración, Loki puede ser usado para probar todos los componentes de la interfaz de usuario. El modo preferido de operación es usando Chrome en un contenedor Docker, evitando así, diferencias de un pixel en la ejecución de pruebas en entornos no idénticos. En nuestra experiencia, las pruebas han sido muy estables pero las actualizaciones de Storybook tienden a hacer que las pruebas fallen con pequeñas diferencias. Parece también imposible probar componentes que usen `“position:fixed”` pero puede ser resuelto al envolver el componente con un `“fixed”.`

Trivy
PROBAR

Construir los flujos que crean e implementan contenedores debe incluir el escaneo de seguridad de contenedores. A nuestros equipos les gusta especialmente Trivy, un escáner de vulnerabilidad para contenedores, gracias a que es más fácil de configurar que otras herramientas y a que se envía como un binario independiente. Ser un software de código abierto y soportar contenedores no distribuidos son otros de los beneficios de Trivy.

Twistlock
PROBAR

Twistlock es un producto comercial con capacidades de detección y prevención de vulnerabilidades de seguridad en tiempo de construcción y tiempo de ejecución. Estas capacidades abarcan la protección de máquinas virtuales, planificadores de contenedores y contenedores en varios registros y repositorios en los que confían las aplicaciones. Twistlock ha ayudado a nuestros equipos a acelerar el desarrollo de aplicaciones reguladas, donde la infraestructura y la arquitectura de las aplicaciones requieren el cumplimiento de, por ejemplo, los estándares de la Industria de Tarjetas de Pago (PCI) y la Ley de Responsabilidad y Portabilidad del Seguro

de Salud (HIPAA). Nuestros equipos han disfrutado de la experiencia de desarrollo que ofrece Twistlock: la capacidad de ejecutar el aprovisionamiento como código, la fácil integración con otras plataformas de observación comunes y los puntos de referencia listos para medir la infraestructura contra las mejores prácticas de consenso en la industria. Ejecutamos Twistlock con análisis de tiempo de ejecución regulares sobre nuestras aplicaciones nativas de la nube, particularmente cuando se requiere algún cumplimiento normativo.

Yocto Project

PROBAR

Un número creciente de dispositivos cada vez más potentes del Internet de las Cosas se ejecutan sobre Linux en vez de sistemas operativos embebidos. Para reducir el uso de recursos y la superficie de ataque, tiene sentido construir una distribución Linux personalizada que solo contenga las herramientas y dependencias necesarias para ejecutar el software en el dispositivo. En este contexto, el Yocto Project ha incrementado su relevancia como herramienta para crear distribuciones Linux a medida para las necesidades de cada caso específico. La curva de aprendizaje es ascendente y dada su falta de flexibilidad, es fácil equivocarse al hacer las cosas. Sin embargo, durante sus muchos años de existencia, el Yocto Project ha atraído una comunidad activa que puede ayudarte. Comparada con otras herramientas similares, es más fácil de integrar en un flujo de Continuous Delivery y, a diferencia de Android Things o Ubuntu Core, por ejemplo, no está ligado a un ecosistema específico.

Aplas

EVALUAR

A menudo, es difícil el controlar los estados del software conforme aumenta su complejidad. Aplas es una nueva herramienta de mapping, que puede usar visualizaciones del software en forma de mapas. La herramienta funciona ingestando metadata sobre el sistema y después muestra un mapa al que se le pueden aplicar diferentes vistas. La ingestión puede ser manual o puede automatizarse a través de APIs. Es interesante ver cómo el producto está evolucionando y observar las posibilidades con la colección automatizada de metadatos. Por ejemplo, es posible mostrar architectural fitness functions como el run cost para crear visualizaciones de cuánto se está gastando en infraestructura en la nube. Otro problema común es entender cómo los sistemas se comunican entre ellos, y a través de esta herramienta lo podemos visualizar.

asdf-vm

EVALUAR

asdf-vm es una herramienta de línea de comandos para administrar múltiples versiones de lenguajes en tiempo de ejecución por proyecto. Es similar a otras herramientas de administración de versiones de línea de comandos, tales como RVM para Ruby y nvm para Node, con la ventaja de una arquitectura extensible basada en plugins para manejar múltiples lenguajes. Su lista actual de plugins incluye muchos lenguajes, además de herramientas tales como Bazel o tflint, cuyas versiones en tiempo de ejecución pueden necesitar ser administradas por proyecto.

AWSume

EVALUAR

AWSume es un script útil para manejar tokens de sesión y asumir credenciales de roles desde la consola de comandos. Encontramos que AWSume es práctico para manejar varias cuentas de AWS al mismo tiempo. En lugar de especificar perfiles individualmente en cada comando, el script lee el caché de la CLI y lo exporta a variables de ambiente. Como resultado, los comandos y los SDKs de AWS escogen las credenciales correctas.

dbt

EVALUAR

La transformación de datos es una parte esencial de los flujos de trabajo de procesamiento de datos: filtrar, agrupar o unir múltiples fuentes en un formato que sea adecuado para el análisis de datos o para alimentar modelos de machine learning. dbt es una herramienta open-source y un producto comercial SaaS que proporciona capacidades de transformación simples y efectivas a los analistas de datos. Los frameworks y herramientas actuales para la transformación de datos, o bien pueden entrar en el grupo de potentes y flexibles — que requieren un conocimiento íntimo del modelo de programación y los lenguajes del framework como es el caso de Apache Spark — o en el grupo de las herramientas con una interfaz de usuario simple tipo drag-and-drop que no se prestan a prácticas fiables de ingeniería como las pruebas y despliegues automatizados. dbt rellena un nicho: usa SQL -una interfaz ampliamente comprendida- para modelar transformaciones simples en lotes, al tiempo que proporciona herramientas de línea de comando que animan a aplicar buenas prácticas de ingeniería como el versionado, las pruebas y el despliegue automatizados; esencialmente implementa modelado de transformación basado en SQL como código. dbt soporta actualmente múltiples fuentes de datos, incluyendo Snowflake y Postgres, y ofrece varias opciones de ejecución, como Airflow y la propia oferta de Apache en la nube. Su capacidad de transformación se limita a lo que ofrece SQL, y no soporta transformaciones de streaming en tiempo real en el momento que escribimos estas líneas.

HERRAMIENTAS

El Yocto Project ha incrementado su relevancia como herramienta para crear distribuciones Linux a medida para las necesidades de cada caso específico.

(Yocto Project)

Aplas es una nueva herramienta de mapping, que puede usar visualizaciones de nuestro software en forma de mapas.

(Aplas)

HERRAMIENTAS

Con una mayor adopción de Kubernetes como orquestador de contenedores, el conjunto de herramientas de seguridad alrededor de contenedores y Kubernetes está evolucionando rápidamente. Falco es una de esas herramientas nativas para contenedores destinadas a abordar la seguridad en tiempo de ejecución.

(Falco)

Docker Notary
EVALUAR

Docker Notary es una herramienta OSS que permite firmar recursos como imágenes, ficheros y contenedores. Esto implica que la procedencia de los recursos puede ser verificada, lo que es muy útil en entornos regulados y una buena práctica en general. Como ejemplo, cuando un contenedor es creado, se le adjunta una firma compuesta de una clave privada y un hash, asociada a la identidad del publicador del contenedor y almacenada en los metadatos del mismo. Una vez publicada, la procedencia del contenedor (o de cualquier otro recurso) puede ser verificada usando su hash y la clave pública del publicador. Las claves suelen estar generalmente disponibles en registros públicos como Docker Trusted Registry, aunque también se pueden publicar en un registro propio. Nuestros equipos han encontrado algunos comportamientos extraños usando servidores Notary locales y sugieren usar un registro que incluya a su vez Notary siempre que sea posible.

Facets
EVALUAR

Dada la creciente cantidad de decisiones importantes derivadas de grandes conjuntos de datos, ya sea directamente o como entrada de entrenamiento para los modelos de machine learning, es importante comprender las diferencias, errores y posibles inconsistencias en sus datos. El proyecto Facets de Google proporciona dos herramientas útiles en este espacio: Facets Overview y Facets Dive. Facets Overview visualiza la distribución de valores para las características en un conjunto de datos, puede mostrar un sesgo de conjunto de entrenamiento y validación y puede usarse para comparar múltiples conjuntos de datos; Facets Dive es para profundizar y visualizar puntos de datos individuales en grandes conjuntos de datos, utilizando diferentes dimensiones visuales para explorar las relaciones entre los atributos. Ambas son herramientas útiles para llevar a cabo pruebas de sesgo ético.

Falco
EVALUAR

Con una mayor adopción de Kubernetes como orquestador de contenedores, el conjunto de herramientas de seguridad alrededor de contenedores y Kubernetes está evolucionando rápidamente. Falco es una de esas herramientas nativas para contenedores destinadas a abordar la seguridad en tiempo de ejecución. Falco aprovecha la instrumentación del kernel de Linux de Sysdig y el perfilaje de llamadas al sistema, permitiéndonos obtener una visión profunda del comportamiento del mismo y ayudándonos a detectar actividades anormales en aplicaciones, contenedores, hosts subyacentes o incluso en el propio orquestador de Kubernetes. Nos agrada la capacidad de Falco para detectar amenazas sin inyectar código de terceros o contenedores adyacentes.

in-toto
EVALUAR

Estamos viendo un mayor uso de la certificación Binaria para asegurar la cadena de suministro de software, particularmente dentro de las industrias reguladas. Los enfoques actualmente preferidos parecen involucrar la construcción de un sistema personalizado para implementar la verificación binaria o confiar en el servicio de un proveedor de la nube. Nos alienta ver la herramienta open-source in-toto entrar en este espacio. In-toto es un framework para verificar criptográficamente cada componente y avanzar hacia el camino a producción de un artefacto de software. El proyecto incluye una serie de integraciones en muchas herramientas de compilación, auditoría de contenedores y despliegue ampliamente utilizadas. Una herramienta de cadena de suministro de software puede ser una pieza crítica del aparato de seguridad de una organización, por lo que nos gusta que, como proyecto open-source, el comportamiento de in-toto sea transparente y la comunidad pueda verificar su propia integridad y cadena de suministro. Tendremos que esperar y mirar si ganará un grupo grande de usuarios y contribuidores para competir en este espacio.

Kubeflow
EVALUAR

Kubeflow es interesante por dos motivos. El primero es su uso innovador de Operadores Kubernetes, el cual habíamos destacado en nuestra edición del Radar de abril 2019. El segundo es que provee una manera de codificar y modelar cargas de trabajo para aprendizaje de máquinas que facilita trasladarlas de un entorno de ejecución a otro. Kubeflow incluye componentes tales como Jupyter notebooks, canalizaciones de datos y herramientas de control. Varios de estos componentes vienen empaquetados como operadores de Kubernetes, para aprovechar su habilidad de reaccionar a los eventos generados por pods mediante la implementación de varias fases de la carga de trabajo. Al tomar los programas individuales y los datos y empaquetarlos como contenedores es posible trasladar cargas de trabajo enteras de un entorno a otro. Esto puede resultar conveniente al trasladar una carga de trabajo útil pero computacionalmente exigente y desarrollada en la nube a una supercomputadora personalizada o a un clúster de unidades de procesamiento tensorial.

MemGuard

EVALUAR

Si tu aplicación gestiona información sensible (tales como claves criptográficas) como texto sin formato en memoria, existe una alta probabilidad de que alguien pueda explotarla como un vector de ataque y comprometer información. La mayoría de las soluciones basadas en la nube a menudo usan módulos de seguridad de hardware (HSM) para evitar estos ataques. Sin embargo, si te encuentras en una situación en la que necesitas hacer esto de forma autónoma sin acceso a un HSM, entonces creemos que MemGuard es bastante útil. MemGuard actúa como un enclave de software seguro para el almacenamiento de información confidencial en la memoria. Aunque MemGuard no reemplaza a los HSM, pero implementa una serie de tácticas de seguridad como la protección contra ataques de arranque en frío, evitando la interferencia con la recolección de basura y fortaleciendo con páginas de protección para reducir la probabilidad de exponer datos confidenciales.

Open Policy Agent (OPA)

EVALUAR

Definir y aplicar políticas de seguridad de manera uniforme en un panorama tecnológico diverso es un desafío. Incluso para aplicaciones simples, se debe controlar el acceso a sus componentes -- tales como orquestadores de contenedores, servicios y los almacenes de datos para mantener el estado de los servicios -- utilizando la configuración de la política de seguridad integrada de sus componentes y los mecanismos de ejecución.
Estamos entusiasmados con Open Policy Agent (OPA), una tecnología de código abierto que intenta resolver este problema.
OPA te permite definir un control de acceso específico y políticas flexibles como código, utilizando el lenguaje de definición de políticas Rego. Rego aplica las políticas de manera distribuida y discreta fuera del código de la aplicación. Al momento de escribir este artículo, OPA implementa una definición y aplicación de políticas uniformes y flexibles para asegurar el acceso a las API de Kubernetes y las API de microservicios a través del sidecar de Envoy y Kafka. También se puede usar como un sidecar para cualquier servicio para verificar las políticas de acceso o filtrar los datos de respuesta. Styra, la compañía detrás de OPA, ofrece soluciones comerciales para una visibilidad centralizada de las políticas distribuidas. Nos gusta ver a OPA madurar a través del programa de incubación CNCF y continuar construyendo soporte para escenarios de cumplimiento de políticas más desafiantes, como diversos almacenes de datos.

Pumba

EVALUAR

Pumba es una herramienta de tests de caos y emulación de redes, para Docker. Pumba puede terminar, detener, quitar o pausar los contenedores de Docker. Pumba también puede emular redes, y simular errores de redes como retrasos, pérdida de paquetes y límites de ancho de banda. Pumba utiliza la herramienta tc para la emulación de redes, lo cual implica que tiene que estar disponible en nuestros contenedores o tenemos que ejecutar Pumba en un contenedor sidecar con tc. Pumba es particularmente útil cuando queremos ejecutar tests de caos automáticamente en un sistema distribuido que se está ejecutando en varios contenedores locales o en un build pipeline.

Skaffold

EVALUAR

Google nos trae Skaffold, una herramienta open-source para automatizar flujos de trabajo de desarrollo locales, incluyendo el despliegue en Kubernetes. Skaffold detecta los cambios en el código fuente y dispara el proceso para construir, etiquetar, y desplegar en un clúster de K8s, incluyendo capturar las trazas y mostrarlas por la línea de comandos. Los flujos de trabajo se pueden enganchar a distintas herramientas de construcción y despliegue, pero esto viene con una configuración dogmática por defecto que hace más fácil comenzar a utilizarla.

What-If Tool

EVALUAR

El mundo del machine learning ha cambiado ligeramente el énfasis de explorar qué modelos son capaces de comprender a cómo lo hacen. La preocupación por la introducción de sesgos o la generalización excesiva de la aplicabilidad de un modelo, ha dado lugar a nuevas herramientas interesantes como What-if Tool (WIT). Esta herramienta ayuda a data scientists a indagar en el comportamiento de un modelo y a visualizar el impacto que varias características y conjuntos de datos tienen en el output. Introducido por Google y disponible a través de Tensorboard o Jupyter notebooks; WIT simplifica las tareas de comparación de modelos, corte de conjuntos de datos, visualización de facets y edición de conjunto de datos individuales. Aunque WIT facilita la realización de estos análisis, todavía requieren una profunda comprensión de las matemáticas y la teoría que hay detrás de los modelos. Es una herramienta para que data scientist obtengan una visión más profunda del comportamiento del modelo. Usuarios ingenuos no deben esperar que ninguna herramienta elimine el riesgo o minimice el daño causado por un algoritmo mal aplicado o mal entrenado.

What-If Tool ayuda a data scientists a indagar en el comportamiento de un modelo y a visualizar el impacto que varias características y conjuntos de datos tienen en el output.

(What-If Tool)

Azure Data Factory para orquestación

RESISTIR

Azure Data Factory (ADF) es actualmente el producto por defecto de Azure, para la orquestación de flujos de procesamiento de datos. Permite ingesta de datos, copia de datos desde y hacia diferentes tipos de almacenamiento tanto On-Prem como Azure, y la ejecución de lógica de transformación. Si bien hemos tenido algunos resultados aceptables con ADF para migraciones simples de almacenes de datos de On-Prem a la nube, desaconsejamos el uso de Azure Data Factory para la orquestación de flujos de procesamiento de datos complejos. Nuestra experiencia ha sido desafiante debido a diferentes factores, incluyendo una cobertura limitada de capacidades que pueden implementarse dando prioridad a la codificación, ya que parece que ADF está priorizando dejar disponible primero las capacidades de plataforma de baja codificación; pocas facilidades para depuración y reporte de errores; observabilidad limitada ya que las capacidades de log de ADF no se integran con otros productos como Azure Data Lake Storage o Databricks, haciendo muy complicado disponer de una observabilidad de extremo a extremo en su lugar; y la disponibilidad de mecanismos de activación de la fuente de datos limitada a algunas regiones solamente.

En este momento, aconsejamos utilizar otras herramientas de orquestación de código abierto (por ejemplo, Airflow) para flujos de datos complejos, y limitar el uso de ADF a copia de datos o snapshots. Confiamos en que ADF solucionará estas deficiencias para poder abastecer adecuadamente más flujos de procesamiento de datos complejos y priorizar el acceso a capacidades que anteponen la codificación.

Azure Data Factory (ADF) es actualmente el producto por defecto de Azure, para la orquestación de flujos de procesamiento de datos.

(Azure Data Factory para orquestación)

LENGUAJES & FRAMEWORKS

Arrow
PROBAR

Arrow es una librería de programación funcional para Kotlin que fue creada mezclando 2 populares librerías ya existentes (kategory y funKTionale). Si bien Kotlin ya contiene bloques nativos para la programación funcional, Arrow provee, además, paquetes de alto nivel de abstracción, listos para ser usados por desarrolladoras/es de aplicaciones. Provee, además, tipos de datos, clases, efectos, ópticos y otros patrones de programación funcional así como integraciones con otras famosas librerías. Nuestras primeras impresiones positivas de Arrow se han visto confirmadas cuando la usamos para construir aplicaciones que ahora están en producción.

Flutter
PROBAR

Varios de nuestros equipos usan Flutter y realmente les gusta. Es un framework multiplataforma que permite escribir aplicaciones móviles nativas en Dart. Se beneficia de Dart y se puede compilar en código nativo y se comunica con la plataforma de destino sin puente y cambio de contexto. La funcionalidad hot-reload de Flutter sigue siendo impresionante y proporciona una respuesta visual súper rápida al editar el código. Estamos seguros de recomendarles que prueben Flutter en uno de sus proyectos.

jest-when
PROBAR

Jest-when es una librería JavaScript ligera que complementa Jest proporcionando los argumentos de llamada de una función simulada. Jest es una gran herramienta para probar el stack, jest-when te permite esperar ciertos argumentos de funciones simulada, y así te permite escribir tests unitarios más robustos para módulos con muchas dependencias.

Micronaut
PROBAR

Micronaut es un framework de la JVM para construir servicios usando Java, Kotlin o Groovy. Se distingue por dejar una pequeña huella en la memoria y por un corto tiempo de arranque. Logra estas mejoras evitando usar reflexión en tiempo de ejecución para la inyección de dependencias (DI) y la generación de proxies, que es un defecto común entre los frameworks tradicionales, y usa, en cambio, un contenedor DI/AOP que realiza la inyección de dependencias en tiempo de compilación. Esto lo vuelve atractivo no solo para microservicios estándar del lado del servidor, sino también en contextos por ejemplo de IoT, aplicaciones Android y funciones serverless. Micronaut usa Netty y tiene soporte de primera clase para programación reactiva. También incluye funcionalidades como service discovery y circuit breaking que lo hace amigable para la nube. Micronaut es un participante muy prometedor para los frameworks full-stack del entorno de la JVM y verlo cada vez en más y más proyectos en producción, nos incita a moverlo a Probar.

ADOPTAR

PROBAR
78. Arrow
79. Flutter
80. jest-when
81. Micronaut
82. React Hooks
83. Biblioteca de Pruebas React
84. Styled components
85. Tensorflow

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

RESISTIR
96. Enzyme

LENGUAJES & FRAMEWORKS

Micronaut es un framework de la JVM para construir servicios usando Java, Kotlin o Groovy. Se distingue por dejar una pequeña huella en la memoria y por un corto tiempo de arranque.

(Micronaut)

La Biblioteca de Pruebas de React es un buen ejemplo de un framework que con un uso más profundo, ha eclipsado las alternativas para convertirse en la opción por defecto más sensata a la hora de probar frontends basados en React.

(React Testing Library)

React Hooks
PROBAR

A principios de este año, React Hooks fueron introducidos en el popular framework de JavaScript. Ellos permiten el uso de state y otras funcionalidades de React sin escribir clases, ofreciendo un procedimiento más claro que otros componentes de higher-order o render-propsp para casos de uso. Librerías como Material UI y Apollo han cambiado a usar Hooks. Hay algunos problemas con el testeo de Hooks, en especial con Enzyme, que contribuyó a reexaminar Enzyme como herramienta favorita.

Biblioteca de Pruebas React
PROBAR

El mundo de JavaScript se mueve bastante rápido, y a medida que adquirimos más experiencia en el uso de un framework, nuestras recomendaciones cambian. La Biblioteca de Pruebas de React es un buen ejemplo de un framework que con un uso más profundo, ha eclipsado las alternativas para convertirse en la opción por defecto más sensata a la hora de probar frontends basados en React. A nuestros equipos les gusta el hecho de que los tests escritos con este framework son menos frágiles que con frameworks alternativos como Enzyme, porque se les anima a probar las relaciones de los componentes individualmente en lugar de probar todos los detalles de implementación.

Styled components
PROBAR

El uso de literales de plantilla etiquetados styled components hace posible colocar el CSS necesario para aplicar estilo a un componente React directamente en el código JavaScript que crea el componente. Esto reduce en gran medida las dificultades en el manejo de CSS y evita la necesidad de convenciones de nombres u otros medios para evitar conflictos de nombres en el CSS. Las personas desarrolladoras pueden ver el estilo al mirar la definición del componente y no tienen que memorizar varios megabytes de CSS. Por supuesto, colocar el CSS en el código JavaScript puede dificultar la obtención de una visión coherente del diseño de diferentes componentes, por lo que recomendamos comprender las consecuencias que tiene utilizar este enfoque.

Tensorflow
PROBAR

Con su versión 2.0, TensorFlow mantiene su prominencia como framework líder en el sector de Machine Learning (ML). TensorFlow comenzó como un paquete de procesamiento numérico que se expandió gradualmente para incluir librerías que permiten varias aproximaciones y entornos de ejecución de ML, abarcando desde las CPUs de los móviles hasta grandes clusters de GPUs. Por el camino, se habilitó una gran cantidad de frameworks para simplificar las tareas de creación de redes neuronales y entrenamiento. Al mismo tiempo, otras frameworks, especialmente PyTorch, ofrecían un modelo de programación imperativa que hacía la depuración y ejecución más simple y fácil. TensorFlow 2.0 sigue ahora por defecto a flujo imperativo (ejecución eager) y adopta Keras como única API de alto nivel. Mientras estos cambios modernizan la usabilidad de TensorFlow y lo hacen más competitivo frente a PyTorch, se trata de una reescritura significativa que a menudo rompe la retrocompatibilidad — muchas herramientas y frameworks de servicio en el ecosistema de TensorFlow no funcionarán de manera inmediata con la nueva versión. De momento, hay que considerar si queremos diseñar y experimentar en TensorFlow 2.0 pero volver a la versión 1 para servir y ejecutar los modelos en producción.

Fairseq
EVALUAR

Fairseq es un conjunto de herramientas de modelado s2s (sequence-to-sequence) desarrollado por Facebook AI Research que permite a investigadoras/es y desarrolladoras/es entrenar modelos personalizados para traducción, síntesis de texto, modelado de lenguaje y otras tareas de procesamiento de lenguaje natural. Es una buena elección para usuarios de PyTorch. Provee referencias de implementación para varios modelos s2s (sequence-to-sequence), admite distribuir entrenamientos a través de múltiples GPUs y máquinas. Es muy extensible y dispone de varios modelos pre-entrenados, incluyendo RoBERTa, una versión optimizada de BERT.

Flair
EVALUAR

Flair es un framework sencillo basado en Python para el procesamiento de lenguaje natural (NLP). Permite a los usuarios realizar tareas estándar de NLP, como reconocimiento de entidades nombradas (NER), part-of-speech tagging (PoS), desambiguación y clasificación del sentido de las palabras y funciona bien en una variedad de tareas de NLP. Flair presenta una interfaz sencilla y unificada para una variedad de embeddings de documentos y palabras, incluyendo BERT, Elmo y las propias de Flair. También tiene soporte multilingüe. El framework en sí, está construido sobre PyTorch. Lo estamos usando en algunos de nuestros proyectos y nos gusta su facilidad de uso y sus potentes abstracciones.

Gatsby.js
EVALUAR

Gatsby.js es un framework para escribir aplicaciones web con una arquitectura conocida como JAMstack. Parte de la aplicación se genera en tiempo de compilación y se despliega como un sitio estático, mientras que el resto de la funcionalidad se implementa como una aplicación web progresiva (PWA) que se ejecuta en el navegador. Dichas aplicaciones funcionan sin código que se ejecute en el lado del servidor. Sin embargo, por lo general, PWA realiza llamadas a APIs de terceros y soluciones SaaS para la gestión de contenido. En el caso de Gatsby.js, todo el cliente y el código de tiempo de compilación están escritos utilizando React. El framework incluye algunas optimizaciones para aumentar la percepción de velocidad de la aplicación web. Proporciona el código y la división de datos out-of the box para minimizar los tiempos de carga y acelera el rendimiento al navegar por la aplicación mediante la captación previa de recursos. Las APIs se llaman a través de GraphQL y varios plugins simplifican la integración con los servicios existentes.

GraphQL
EVALUAR

Hemos visto muchas implementaciones exitosas de GraphQL en nuestros proyectos. También hemos visto algunos patrones de uso interesantes, incluyendo GraphQL for server-side resource aggregation. Dicho esto, existen inquietudes sobre el mal uso de este framework y algunos de los problemas que pueden ocurrir. Por ejemplo problemas de rendimiento alrededor de consultas N+1 y código repetitivo necesario cuando se agregan nuevos modelos, conduciendo a complejidad. Hay alternativas a estos problemas como el cacheo de consultas. Sin embargo, no es una "bala de plata", todavía pensamos que vale la pena una evaluación como parte de tu arquitectura.

KotlinTest
EVALUAR

KotlinTest es una herramienta de prueba independiente para el ecosistema de Kotlin que les ha gustado a nuestros equipos. Permite pruebas unitarias basadas en propiedades, una técnica que hemos resaltado anteriormente en el Radar. Las ventajas clave son que ofrece una variedad de estilos de pruebas para estructurar los conjuntos de pruebas y que posee un conjunto completo de matchers, lo que permite pruebas expresivas en un elegante lenguaje específico de dominio (DSL) interno.

NestJS
EVALUAR

NestJS es un framework Node.js del lado del servidor escrito en TypeScript. Al integrar la riqueza de la comunidad de Node.js, NestJS proporciona una arquitectura de aplicación lista para usar. El modelo mental para desarrollar NestJS es similar a la versión del lado del servidor de Angular o a la versión TypeScript de Spring Boot, por lo que la curva de aprendizaje para las personas desarrolladoras es baja. NestJS soporta protocolos como GraphQL, Websocket y bibliotecas ORM.

Paged.js
EVALUAR

Cuando utilizamos HTML y tecnologías similares para la creación de libros y de otros tipos de publicaciones, se debe tener en cuenta la paginación. Incluyendo contadores de página, elementos repetidos en cabeceras y pies de página, así como mecanismos que evitan molestos saltos de página. Paged.js es una librería open-source que implementa una serie de polyfills para los módulos de CSS Paged Media y Generated Content for Paged Media. Todavía es experimental pero cubre un gap importante en el "escribe una vez, publica en todas partes" historia del HTML.

LENGUAJES & FRAMEWORKS

Gatsby.js es un framework para escribir aplicaciones web con una arquitectura conocida como JAMstack. Parte de la aplicación se genera en tiempo de compilación y se despliega como un sitio estático, mientras que el resto de la funcionalidad se implementa como una aplicación web progresiva (PWA) que se ejecuta en el navegador

(Gatsby.js)

LENGUAJES & FRAMEWORKS

Quarkus es un framework basado en contenedores y nativo de la nube de Red Hat para escribir aplicaciones en Java. Tiene un tiempo de arranque muy corto (decenas de milisegundos) y tiene utilización de memoria baja

(Quarkus)

Quarkus
EVALUAR

Quarkus es un framework basado en contenedores y nativo de la nube de Red Hat para escribir aplicaciones en Java. Tiene un tiempo de arranque muy corto (decenas de milisegundos) y tiene utilización de memoria baja lo que lo convierte en un buen candidato para FaaS o para escalado hacia arriba o hacia abajo de manera frecuente en un orquestador de contenedores. Al igual que Micronaut, Quarkus consigue esto utilizando técnicas de compilación ahead-of-time para realizar la inyección de dependencias en tiempo de compilación y evitar los costes de la reflexión en tiempo de ejecución. También funciona bien con la Imagen Nativa de GraalVM que reduce aún más el tiempo de arranque. Quarkus soporta los modelos tanto imperativo como reactivo. Junto a Micronaut y Helidon, Quarkus lidera la carga en la nueva generación de frameworks Java que tratan de solucionar el problema del tiempo de arranque y consumo de memoria sin sacrificar la efectividad de la persona que desarrolla. Ha recibido mucha atención por parte de la comunidad y merece la pena mantenerla bajo vigilancia.

SwiftUI
EVALUAR

Apple ha dado un gran paso adelante con su nueva framework SwiftUI para implementar interfaces de usuario en plataformas macOS e iOS. Nos gusta que SwiftUI se mueva más allá de la relación parche entre Interface Builder y XCode, y adopte una aproximación coherente, declarativa y centrada en el código. Ahora puedes ver tu código y la interfaz visual resultante una al lado de la otra en XCode 11, proporcionando una experiencia mucho mejor para la persona desarrolladora. La framework SwiftUI también se ha inspirado en el mundo React.js que ha venido dominando el desarrollo web durante los últimos tiempos. Los valores inmutables en los modelos de vista y un mecanismo de actualización asíncrono proporcionan un modelo unificado de programación reactiva. Esto proporciona a las/os desarrolladoras/es una alternativa totalmente nativa a la proporcionada por frameworks reactivos similares como React Native o Flutter. Aunque SwiftUI representa definitivamente el futuro del desarrollo de Apple UI, es bastante nueva y llevará tiempo limar sus bordes ásperos. Estamos deseando ver documentación mejorada y una comunidad de personas desarrolladoras que puedan establecer un conjunto de prácticas para pruebas y otros aspectos de ingeniería. and a community of developers who can establish a set of practices for testing and other engineering concerns.

Testcontainers
EVALUAR

Crear entornos confiables para la ejecución de tests automatizados es un problema recurrente, particularmente dado que el número de componentes de los que dependen los sistemas modernos sigue incrementándose. Testcontainers es una librería Java que ayuda a mitigar este reto gestionando dependencias dockerizadas para tus tests. Esto es particularmente útil para gestionar repetidamente instancias de bases de datos o infraestructuras similares, pero también puede ser usado en navegadores web para tests de UI. Nuestros equipos han encontrado esta librería útil para realizar tests de integración más fiables con estos contenedores programables, ligeros y desechables.

Enzyme
RESISTIR

No solemos mover herramientas obsoletas a “Hold” en el Radar. Sin embargo, nuestros equipos creen firmemente que Enzyme ha sido reemplazado por React Testing Library en el campo del Unit Testing para componentes React UI. Los equipos que usan Enzyme han visto que esta herramienta se centra en el testing de las partes internas de los componentes haciendo los test frágiles difíciles de mantener.

¿Quieres estar al día con todas las noticias e insights del Radar?
Síguenos en tu red
social favorita o suscríbete.

suscríbete ahora

ThoughtWorks®

Somos una consultora de software global y una comunidad de individuos apasionados guiados por propósitos. Pensamos 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 7000 personas, incluyendo una división de productos que crea herramientas pioneras de software para equipos. ThoughtWorks cuenta con 43 oficinas en 14 countries: Australia, Alemania, Brasil, Canada, Chile, China, Ecuador, España, Estados Unidos, India, Italia, Reino Unido, Singapur yTailandia

thoughtworks.com

ThoughtWorks®

thoughtworks.com/es/radar

#TWTechRadar