ThoughtWork®

NOVEDADES

A continuación, se presentan las tendencias destacadas en esta edición:

INCORPÓRESE AL MUNDO JAVASCRIPT

Creíamos que la tasa de cambio en el espacio de código abierto de Ruby era rápida hasta que llegaron los frameworks veloces de JavaScript. JavaScript solía ser una tecnología de condimento, utilizada siempre para acrecentar otras tecnologías. Si bien aún mantiene esa función, ahora amplió su rol en su propia plataforma con una tasa de cambio asombrosa. Intentar entender la magnitud de este espacio es totalmente abrumador y la innovación es increíble. Al igual que los espacios de códigos abiertos Java y Ruby, esperamos que en algún momento se calme, al menos un poco.

LEY DE CONWAY

La ley de Conway, que establece que “las organizaciones que diseñan sistemas ... se limitan a elaborar diseños de aplicaciones que son copias de las estructuras de comunicación de dichas organizaciones”, sigue apareciendo en lugares inesperados. Uno de los principios clave del Agile Manifesto es “Las personas por sobre los procesos y las herramientas” y podemos ver cómo la ley de Conway refuerza esta idea, de un modo positivo aunque también negativo. Algunas empresas están inmersas en estructuras de silos que agregan una fricción innecesaria a los esfuerzos de ingeniería, al mismo tiempo que otras empresas más informadas usan la organización de equipos para impulsar los tipos de arquitectura que desean. Estamos en medio del aprendizaje sobre los peligros que corremos si

obtenemos si la aprovechamos.

LOS MICROSERVICIOS Y EL AUGE DE API

Podemos observar un gran interés en la arquitectura de los microservicios y también una especial atención sobre la importancia de la API dentro de una organización y como un puente hacia el mundo real. En una arquitectura de microservicios, gran cantidad de servicios muy pequeños se implementan y están vinculados a la creación de sistemas, con los servicios asociados estrechamente a conceptos y valores comerciales. Para lograr que este enfoque funcione, los equipos necesitan una buena disciplina en torno a la creación, pruebas, integración y, luego, gestión de los servicios. Esta edición del Radar analiza algunas

microservicios.

NUEVA DESCENTRALIZACIÓN

Internet comenzó su historia como un sistema distribuido, pero durante la última década o más, hemos visto una gran centralización de servicios y

correos electrónicos de todo el mundo se transmiten solamente a través de 10 proveedores. Algo similar sucede con la computación en la nube, ya que solo una reducida cantidad de proveedores brindan servicios a la gran mayoría de nuestras necesidades en la nube. Podemos observar una nueva tendencia de “volver a descentralizar” los datos y la infraestructura, impulsada en parte por las revelaciones acerca del dominio de los

por el deseo de mantener un control más individual y organizativo.

ABOUT THE TECHNOLOGY RADAR

Las personas que trabajamos en ThoughtWorks sentimos pasión por la tecnología. La creamos, la investigamos, la probamos, la liberamos, escribimos acerca de ella y siempre buscamos mejorarla, para todos. Nuestra misión es defender la excelencia en software y revolucionar el mundo de las tecnologías de la información (TI). Por eso creamos y compartimos el Radar Tecnológico de ThoughtWorks como respaldo a esa misión. La creación del radar es obra de la Comisión Asesora de Tecnología de ThoughtWorks, un grupo de líderes sénior en tecnología de ThoughtWorks. Los miembros de este equipo se reúnen en forma periódica para debatir acerca de la estrategia global en materia de tecnología de ThoughtWorks y abordar las tendencias tecnológicas que repercuten en gran medida sobre nuestra industria.

El radar captura el resultado de los debates de la Comisión Asesora de Tecnología en un formato que aporta valor a una amplia variedad de partes interesadas, desde los directores de información hasta los desarrolladores. El contenido tiene el propósito de brindar un resumen conciso. Lo alentamos a explorar estas tecnologías para

Con f dencialidad Directa Perfecta (Perfect Forward

sesiones de comunicaciones previas, incluso si después las llaves maestras del servidor se ven comprometidas. A pesar de que habilitar las conexiones HTTPS es muy

manera y se recomienda habilitar el PFS para aumentar la seguridad.

A medida que las aplicaciones JavaScript del lado del

vayan a la par. Una falla común en la arquitectura es el acceso sin restricciones al Modelo de Objetos del Documento (DOM, por sus siglas en inglés) a través de la base de código (al combinar la manipulación del DOM con la lógica de las aplicaciones y las llamadas AJAX). Esto hace que el código sea difícil de entender y extender. Pensar en la separación de responsabilidades es un antídoto muy útil, ya que limita en forma estricta todos los accesos al DOM (lo cual, por lo general, se traduce en el uso de jQuery) a una capa delgada de segregación. Un efecto secundario positivo que resulta de este enfoque es que todo lo que quede fuera de esta capa de **segregación del DOM** puede probarse de manera rápida y aislada desde el navegador mediante el uso de un motor JavaScript como **node.js**.

Cuando se utilizan técnicas tales como “instrumentar todas las cosas” y el registro semántico, puede resultar de gran ayuda la **captura explícita de los eventos de dominios**. Usted puede evitar tener que inferir cuáles son las intenciones del usuario detrás de las transiciones

Event sourcing garantiza que todos los cambios que se realicen al estado de la aplicación se almacenen como una secuencia de eventos. No solo podemos buscar estos eventos, sino que también podemos usar el registro de eventos para reconstruir estados anteriores y además, podemos utilizarlo como base para ajustar de forma automática el estado para enfrentar cambios retroactivos. Además de la selección

tiene implicaciones positivas para el análisis de cómo conseguir mayor conocimiento sobre el cliente.

En las organizaciones DevOps-savvy, generalmente los

producción y ellos mismos responden a los incidentes. Gracias a esta visibilidad y al acceso a los entornos de

Las tendencias tecnológicas han derrumbado los muros que antes rodeaban las redes de tecnología informática empresarial y esto derivó en una **empresa sin límites**. Con frecuencia, los empleados usan sus propios dispositivos de cliente para acceder a los datos corporativos a través de servicios en la nube y API para web y, por lo general, sin el conocimiento de la organización'. Debido a que los dispositivos se siguen multiplicando y cada vez más aplicaciones se ubican en la nube, los negocios se ven obligados a pensar nuevamente acerca de suposiciones fundamentales sobre el acceso a los datos y la seguridad de las redes.

La virtualizacion de servidores y la computacion en la nube han hecho sencillo el conseguir y abastecer hardware y servidores virtuales. Sin embargo, esta

administrar nuestros bienes virtuales se ha vuelto cada vez más complejo. El uso de técnicas más familiares con el mundo del desarrollo de software, tales como TDD, BDD y CI, ofrece un enfoque para manejar esta

de efectuar cambios en nuestra infraestructura en forma segura, repetible y automatizada. Las herramientas para **pruebas de aprovisionamiento, como rspec-puppet, Test Kitchen y severspec, estan disponibles para la mayoria de las plataformas**.

Con la proliferación de las aplicaciones JavaScript de una sola página, hemos encontrado que las llamadas Ajax lentas, la manipulación excesiva del DOM y los errores JavaScript inesperados en el Navegador pueden tener un gran impacto en la capacidad de respuesta de un sitio web. Es muy importante recopilar y añadir

monitorización de los usuarios reales proporciona una advertencia y diagnóstico anticipados de problemas de producción,

situación en particular. http://newrelic.com/real-user-monitoring

En el último radar hablamos acerca de la Captura Explícita de los Eventos de Dominio, con una especial atención centrada en el registro de los eventos

transiciones de estado y no solamente de entidades CRUD. A pesar de que es común que las interfaces REST usen PUT para actualizar el estado de los recursos, generalmente POST es mejor para registrar el recurso de un nuevo evento, lo que captura la intención. **REST sin PUT** tiene la ventaja secundaria de separar interfaces de comandos y consultas, forzando a los consumidores del servicio a una consistencia eventual.

El hecho de considerar los logs en términos de datos nos brinda una perspectiva más amplia en cuanto a la actividad operativa de los sistemas que creamos. El **logging estructurado**, que consisten en el uso de un formato de mensaje consistente y predeterminado que contenga información semántica, se construyen sobre esta técnica y permiten a las herramientas como Greylog2 y Splunk plantear visiones más profundas.

Observamos múltiples organizaciones que crean una **Plantilla de Servicio Adaptada** que puede utilizarse para generar rápidamente nuevos servicios, pre-

producción de una organización. La plantilla contiene un conjunto de decisiones predeterminadas, tales como: frameworks web, registros, monitoreo, construcción, empaquetamiento y enfoques de despliegue. Esta es una técnica muy útil que impulsa la evolución colaborativa, manteniendo una gobernabilidad ligera.

La reducción de costos, el tamaño, el consumo de energía y la simplicidad de los dispositivos físicos han generado una explosión en los dispositivos que abren los dominios físicos al software. Por lo general, estos dispositivos no contienen mucho más que un sensor y un componente de comunicación, como Bluetooth Low Energy o WiFi. Como ingenieros de software, necesitamos ampliar nuestras ideas para incluir **la unión del mundo físico con el digital mediante hardware simple**. Ya podemos notar la presencia de este fenómeno en el auto, el hogar, el cuerpo humano, la agricultura y otros entornos físicos. El tiempo y los costos necesarios para realizar un prototipo de estos dispositivos disminuyen para ajustarse a las rápidas iteraciones posibles en software.

En nuestro afán por respaldar los modelos de negocio que cambian constantemente, aprender de las conductas pasadas y brindar la mejor experiencia para cada uno de los visitantes, sentimos la tentación de querer grabar la mayor cantidad de datos posible. Al mismo tiempo, los hackers están más feroces que nunca y protagonizan impresionantes violaciones de la seguridad sin descanso. A su vez, ahora nos enteramos de la existencia de una vigilancia masiva sin precedentes por parte de las agencias gubernamentales. El término **Datensparsamkeit** proviene de la legislación alemana en materia de privacidad y describe la idea de almacenar tanta información personal como sea absolutamente necesaria para la empresa o las leyes pertinentes. Algunos ejemplos de esto son, en lugar de almacenar la dirección IP completa del cliente en los registros de acceso, utilizar solamente los primeros dos o tres octetos y, en vez de registrar trayectos de tránsito con un nombre de usuario, utilizar un símbolo anónimo. Si nunca almacena la información, ya no tendrá que preocuparse de que alguien quiera robársela.

Muchos despliegues requieren imágenes de máquinas para distintos roles de un servidor, aplicaciones y servicios, base de datos, proxy inversos, etc. Debido a que la creacioón de una imagen de una máquina desde cero, mediante el uso del ISO de un sistema operativo y scripts de provisionamiento, puede tardar mucho tiempo, puede ser muy útil crear un ***pipeline* de imágenes de máquinas**. La primera etapa en el pipeline establece una imagen base de acuerdo con los estándares generales de la organización. Las etapas posteriores pueden mejorar la imagen base

tienen requisitos similares (por ejemplo, un servidor de aplicaciones), el pipeline puede extenderse mediante una etapa intermedia, que toma la imagen base y proporciona una imagen con un servidor de aplicaciones pero sin aplicaciones ni servicios. Estos pipelines no son

imagen de base.s

De todos los enfoques con los que podemos no estar de acuerdo, equiparar velocidad con productividad se ha convertido en un tema tan frecuente que pensamos que teníamos que abordarlo en nuestro círculo de espera. Cuando se la utiliza correctamente, la velocidad permite la incorporación del “clima de ayer” en el proceso de

la capacidad estimada para un equipo dado en un momento dado. Puede mejorar a medida que los miembros del equipo empiezan a integrarse o al arreglar problemas como las deudas técnicas o un servidor que ha dejado de responder. Sin embargo, como cualquier otra métrica, la velocidad puede utilizarse de forma incorrecta. Por ejemplo, los gerentes de proyectos que son demasiado entusiastas tienden a insistir en la mejora continua de la velocidad. El hecho de considerar a la **velocidad en términos de productividad** genera conductas de equipo improductivas que optimizan la métrica a costa del funcionamiento real del software.

La arquitectura inicial de Hadoop se basó en el paradigma del escalamiento de datos de forma horizontal y de metadatos de forma vertical. A pesar de que los nodos esclavos administraban el almacenamiento y procesamiento de datos bastante bien, los nodos maestros que gestionaban los megadatos representaban un punto único de falla y limitaban el uso web en escala. **Hadoop 2.0**

PLATAFORMAS *continuación*

La cantidad y la madurez de las opciones de **nubes privadas** en las instalaciones continúan creciendo. Las soluciones que van desde las opciones basadas en OpenStack (como la nube privada de Rackspace) hasta las opciones PAAS (como CloudFoundry) son aquellas que deberían considerar las organizaciones que buscan utilizar la infraestructura existente o las que necesitan un mayor nivel de control sobre la nube fuera de las instalaciones.

Recientemente, AMD lanzó un **Soc (sistema en chip) ARM** de 8 núcleos diseñado para servidores y

integrados en 2015. Los servidores basados en ARM representan una alternativa interesante de x86 por

energía. Para determinados niveles de procesamiento, es preferible la creación de una nube impulsada por ARM. http://www.anandtech.com/show/7989/amd-

SPDY es un protocolo de red abierto para el transporte de baja latencia de contenido web propuesto para HTTP 2.0 que ha observado un aumento en la compatibilidad de navegadores modernos. SPDY reduce el tiempo de carga de páginas al priorizar la transferencia de subrecursos, de modo que solo se necesita una conexión por cliente. La seguridad de la capa de transporte se utiliza en las implementaciones SPDY con los

Al tiempo que la integración centralizada de datos para la elaboración de análisis e informes continúa siendo una buena estrategia, las iniciativas tradicionales del **Data Warehouse Empresarial** (EDW, por sus siglas en inglés) presentan una tasa de fallos superior

adelantado traen, como consecuencia, almacenamientos sobredimensionados que tardan años en completarse y son muy costosos de mantener. En esta edición del radar, dejamos estas técnicas y EDW antiguos en espera. En cambio, recomendamos evolucionar hacia un EDW. Pruebe y aprenda al lograr incrementos pequeños pero valiosos que con frecuencia se liberan a la producción. Las herramientas y técnicas no tradicionales pueden ayudar. Por ejemplo, se puede utilizar un diseño de esquema Data Vault o incluso un almacenamiento de documentos NoSQL como HDFS.

OSGi (del inglés Open Service Gateway initiative) es una

modular para Java, al permitir la recarga dinámica de componentes. A pesar de que algunos proyectos (en especial Eclipse) utilizan OSGi correctamente, otros han expuesto los riesgos de agregar abstracciones a plataformas que nunca se diseñaron para esos usos.

sistema de componentes, rápidamente advierten que esto solo soluciona una parte de todo el problema y, por lo general, agrega su propia complejidad accidental a los proyectos, así como a creaciones más complejas. En la actualidad, la mayoría de los proyectos o bien usan archivos JAR pasados de moda o arquitecturas de microservicios para gestionar los componentes, al tiempo que esperan la solución nativa de Java en la

HERRAMIENTAS

Desde la primera presentación de **Ansible** en el último radar, no dejamos de asombrarnos acerca de sus capacidades y facilidad de uso, en comparación con otras ofertas en este espacio. Si nos basamos en nuestras experiencias durante el último año, no dudamos en recomendar Ansible como una gran opción para el control automatizado de su infraestructura.

El uso de herramientas de **gestión de dependencias de JavaScript** ha ayudado a nuestros equipos de entrega a manejar grandes cantidades de JavaScript estructurando su código y cargando las dependencias en

esfuerzos en la mayoría de los casos, las cargas diferidas complican la compatibilidad con el modo fuera de línea. Las diferentes herramientas de gestión de dependencias cuentan con distintas fortalezas, por lo que deberá elegir según su contexto.

CartoDB es una herramienta GIS (sistema de

código abierto, basada en PostGIS y PostgreSQL. Permite el almacenamiento y la búsqueda de datos geoespaciales a través de SQL. Además, ofrece una práctica biblioteca JavaScript, CartoDB.js, para representar los mapas y visualizar los datos.

Según nuestra recomendación del último radar acerca de enfocarse en la reducción del tiempo medio de recuperación, deseamos destacar a **Chaos Monkey** de la

que desactiva en forma aleatoria determinados casos en el entorno de producción durante la operación normal. Cuando se ejecuta acompañada de un monitoreo integral y el respaldo de un equipo, ayuda a descubrir debilidades inesperadas en el sistema. A su vez, esto le permite al equipo de desarrollo construir mecanismos de recuperación automática con antelación en lugar de

HERRAMIENTAS *continuación*

Las migraciones de bases de datos automatizadas son comunes en los proyectos ágiles y estamos muy contentos de ver avances en las herramientas para este espacio. **Flyway**

HERRAMIENTAS *continuación*

Siempre hemos defendido el uso de prototipos hechos

del usuario sin quedar atrapado en los detalles del **Prototype On Paper** es una herramienta que permite capturar las maquetas individuales elaboradas en papel a través de una cámara con iOS o Android y vincularlas para permitir las pruebas de interacción del usuario. Esto acorta la distancia entre los

Protractor es un marco de prueba basado en Jasmine que envuelve a WebDriverJS con una funcionalidad

Angular.JS. Descubrimos que es excelente en cuanto a la rápida evolución del espacio de frameworks de prueba JavaScript. A pesar de que se diseñó para ejecutar pruebas de extremo a extremo con un backend real, las pruebas de Protractor también pueden realizarse para trabajar con una puerta de entrada HTTP con servicio de stub para ejecutar solo pruebas del lado del cliente.

En la última edición del Radar, mencionamos **SnapCI** de ThoughtWorks (un servicio alojado que brinda herramientas de implementación). Desde ese momento, hemos observado que muchos equipos utilizan SnapCI con éxito en sus proyectos. Si necesita una solución de entrega simple y constante en la nube, SnapCI puede brindársela con un solo clic. Sin hardware, sin problemas.

Dado el creciente control de la privacidad de los datos, cada vez son más las empresas que se preocupan a la hora de compartir los análisis web con terceros. **Snowplow Analytics y Piwik** son ejemplos de plataformas de análisis de código abierto que pueden alojarse a sí mismas y proporcionar un conjunto de funciones y planes de trabajo prometedores.

La creciente complejidad en las aplicaciones web ha aumentado la concientización respecto de que la apariencia también debe ser probada además de la funcionalidad. Esto ha provocado el surgimiento de una variedad de **herramientas de pruebas de regresión visual**, incluidas CSS Critic, dpxdt, Huxley, PhantomCSS

directas de valores CSS hasta las comparaciones reales de las capturas de pantalla. A pesar de que este en un campo que aún se encuentra en desarrollo activo, creemos que las pruebas de regresión visual deben añadirse a las formas de entrega continua.

Cada vez es más importante la automatización de pruebas para dispositivos móviles. **Appium** es un framework de automatización de pruebas que puede utilizarse para probar la web móvil y aplicaciones móviles nativas e híbridas en iOS y Android. En su parte central, Appium es un servidor web que expone un REST API, recibe conexiones de un cliente y escucha los comandos, ejecuta esos comandos en un dispositivo móvil y contesta con HTTP que representa el resultado de la ejecución del comando. Permite que las pruebas puedan escribirse en múltiples plataformas (iOS, Android) mediante el uso de la misma API. Appium es de

uso de npm. (www.appium.io)

Consul hace los servicios se registren a sí mismos fácilmente y descubran otros servicios vía DNS o HTTP. Escala de forma automática, con una búsqueda de servicios de manera local o en todos los centros de datos. Además, Consul proporciona un

que utiliza Consul está impulsado por la biblioteca Serf, que aprovecha y construye sobre las características de detección de fallos y membresía. (http://www.consul.io, http://www.serfdom.io)

Cuando se usan técnicas como “instrumentar todas las cosas” y el registro semántico, es posible que termine con una gran cantidad de datos de registros. La compilación, consolidación y movimiento de estos datos puede resultar un verdadero problema y **Flume** es un sistema distribuido que justamente aborda esos

para HDFS, Flume puede mover fácilmente enormes cantidades (varios terabytes) de datos de registros desde múltiples orígenes hacia un almacenamiento centralizado de datos para su procesamiento posterior.

Todos los avances para iOS deben llevarse a cabo en OS X. Debido a las restricciones técnicas y de licencias, la ejecución de granjas de servidores con OS X no es

Travis CI, con la ayuda de Sauce Labs, ahora brinda servicios de integración continua basados en la nube para proyectos iOS y OS X.

LENGUAJES Y FRAMEWORKS

Dropwizard es una combinación sólida y práctica de varias herramientas y marcos de trabajo ligeros Java, muchos de los cuales tienen méritos por su cuenta. El paquete incorpora muchas de nuestras técnicas favoritas, entre ellas un servidor HTTP integrado, soporte para puntos de acceso RESTful, métricas operativas y controles de salud integrados y además, la capacidad de poner en marcha la aplicación con facilidad. Hacer las cosas bien es muy simple con Dropwizard, ya que permite enfocarse en resolver los problemas "de negocio" en lugar de preocuparse mucho por la infraestructura.

El **lenguaje Go** ha pasado gradualmente de ser "Simplemente un Lenguaje Más" a convertirse en una valiosa herramienta para muchos proyectos. A pesar de ser resueltamente de paradigma simple en un mundo de lenguajes cada vez más complejos, parece mantener un buen equilibrio entre expresividad, potencia y simplicidad.

El equipo que se encuentra detrás de **Java 8** tuvo que luchar dos batallas: las fuerzas de la comunidad siempre a favor de la retrocompatibilidad (un sello tradicional de Java) y el desafío técnico de llevar a cabo un cambio profundo en las bibliotecas y funciones existentes. Triunfaron en ambos frentes, ya que le dieron una nueva vida al lenguaje Java y lo ubicaron a la par de otros lenguajes tradicionales en cuanto a características funcionales de programación. En particular, Java 8 cuenta con una magia sintáctica excepcional que permite una interoperabilidad perfecta entre los bloques Lambda, la nueva característica conocida como funciones de orden superior, y las interfaces SAM (del inglés Single Abstract Method), la forma tradicional de transmitir comportamiento.

La **arquitectura reactiva** se sigue extendiendo a lo largo de las plataformas y los paradigmas, simplemente porque soluciona un problema común de una forma elegante y así logra ocultar las inevitables conexiones de la aplicación en un agradable aislamiento.

Scala es un lenguaje de gran tamaño que es popular gracias a su accesibilidad para los desarrolladores nuevos. Este abanico de características es un problema, ya que muchos aspectos de Scala, como las conversiones implícitas y el dinamismo, pueden generar inconvenientes. Para utilizar Scala de manera exitosa, es preciso investigar el lenguaje y formar una opinión sólida acerca de cuáles son las partes adecuadas para usted **las partes buenas de Scala**. Puede desactivar las partes que no desee mediante el uso de un sistema denominado "banderas de funcionalidades".

Seguimos viendo a los frameworks JavaScript como una buena manera para estructurar código y aportar mejores técnicas de programación a JavaScript. **AngularJS** se utiliza en gran medida en los proyectos de ThoughtWorks. Sin embargo, siempre les aconsejamos a los equipos que evalúen otras buenas alternativas, como Ember.js y Knockout.js.

Julia es un lenguaje de programación dinámico, de procedimiento y homoicónico que fue diseñado con el objetivo de cubrir las necesidades de computación

lenguaje se organiza en torno al concepto de funciones genéricas y métodos de distribución dinámicos. Los programas Julia son, en su mayoría, funciones que

combinaciones de tipos de argumentos. La combinación de estas características del lenguaje y el compilador en tiempo real basado en LLVM contribuyen a que Julia alcance un rendimiento de alto nivel. Además, Julia admite un entorno de multiprocesamiento basado en el intercambio de mensajes que permite que los programas se ejecuten en procesos múltiples. Esto les permite a los programadores crear programas distribuidos basados en cualquiera de los modelos de programación paralela.

La adopción de todo el conjunto Clojure (los lenguajes Clojure y ClojureScript y, como una opción adicional, la base de datos Datomic) proporciona algunas ventajas, como las estructuras de datos inmutables que van desde la interfaz del usuario hasta el backend. Han aparecido muchos frameworks en el espacio Clojure para aprovechar estas características únicas, pero hasta el momento la más prometedora es **Om**. Om es una cubierta de ClojureScript en torno al marco de

ThoughtWorks – Es una consultora global, empresa de productos de software y una comunidad de personas apasionadas cuyo propósito es revolucionar el desarrollo y creación de software, promoviendo impacto social positivo en los países y comunidades donde actua. Su división de productos, ThoughtWorks Studios, desarrolla herramientas pioneras para equipos de software - tales como Mingle ®, Go ™ y Twist ®, y que ayudan a las organizaciones a colaborar y entregar software de calidad.

Los clientes de ThoughtWorks son organizaciones com misiones ambiciosas que buscan abordajes y tecnologías innovadoras como forma de alcanzar sus objetivos. Con 20 años de experiencia en el mercado, ThoughtWorks tiene más de 2.500 empleados - los “ThoughtWorkers” -

Australia, Brasil, Canada, China, Estados Unidos, Ecuador India, Inglaterra, Singapur y Uganda.

ThoughtWorks®