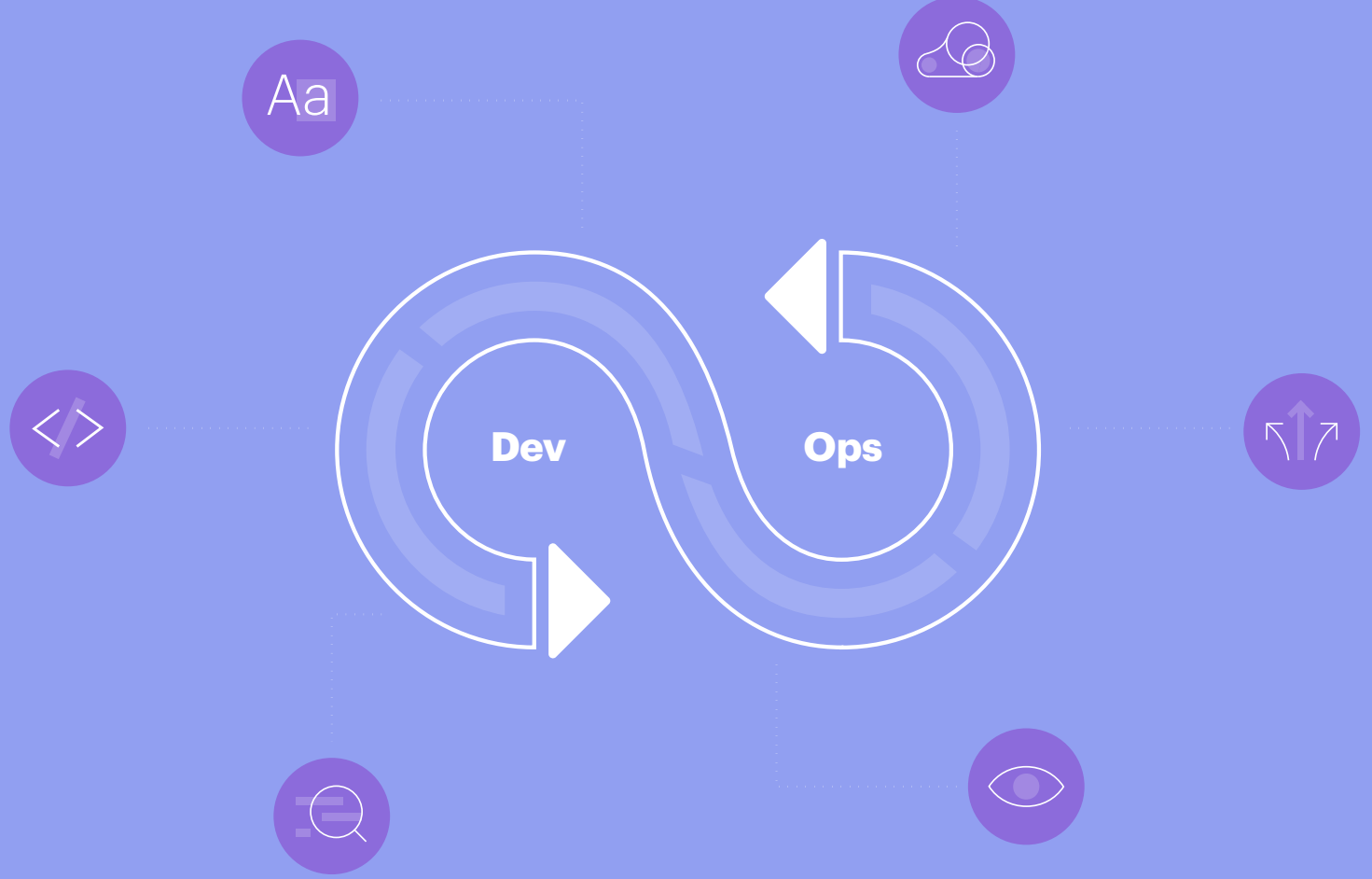
DevOps:
guía de implementación

Whitepaper



GeneXus™
by Globant

El mercado actual exige que los desarrollos y la liberación de funcionalidades de software sean cada vez más ágiles.

Para poder cumplir con esa entrega continua de valor, es necesario evaluar y hacer cambios en los procesos, las herramientas y las formas de trabajo a través de una metodología, propia de la ingeniería de software, conocida como DevOps.

El término, que es una combinación de las palabras "development" (desarrollo) y "operations" (operaciones), fue inventado por el programador y consultor de tecnología de origen belga, Patrick Debois, para resumir los beneficios que se obtienen al eliminar las barreras entre los equipos de desarrollo y operaciones, a través de la implementación de herramientas y procesos ágiles.

La palabra tomó relevancia mundial en el año 2009 tras el lanzamiento del evento Devopsdays (organizado por Debois), alcanzando repercusiones en las redes sociales, y llamando la atención de grandes de la industria como IBM.

A continuación detallamos cuáles son los cambios que deben realizar las organizaciones que deseen implementar una cultura DevOps, y cuáles son los procesos y herramientas que se deben utilizar para lograr la entrega de un mejor producto.

¿Por qué iniciarse en el mundo DevOps?

Las empresas de tecnología conocen la importancia de liberar sus productos al mercado de forma ágil y continua, pero les resulta difícil encontrar el mejor camino para resolver situaciones como:

- ☑ Las barreras que hay entre los distintos equipos de trabajo.
- ☑ La fluidez entre las colaboraciones internas.
- ☑ El aprovechamiento de los recursos disponibles.
- ☑ La agilización de la producción de aplicaciones.
- ☑ La reducción de los tiempos de fallas de los servicios.
- ☑ La incrementación de las automatizaciones.
- ☑ La seguridad informática.

DevOps es la forma de trabajo que ayuda a resolver muchos de estos puntos.

A través de un conjunto de prácticas, esta metodología plantea una nueva forma de pensar y de encarar las situaciones para reducir las barreras entre las personas y mejorar la automatización de los procesos de todas las etapas de la cadena de valor.

«DevOps está enfocado en principios, prácticas y resultados. Para hacer una movida hacia DevOps, es muy importante apostar por preparar a los equipos, entender todo lo que hay alrededor de esta cultura, ajustar los procesos a lo que realmente se está necesitando, y por último, instalar la tecnología que ayude a la implementación de esas prácticas»

Genry Leiva. Consultor DevOps, en el webinar Haciendo DevOps con GeneXus, ofrecido con Enrique Almeida, Ingeniero de Software.

En una entrevista realizada por Devopslatam, Debois detalló que la clave para lograr la integración de los equipos radica en comprender los problemas que puedan tener los compañeros de las otras áreas. “A veces pensamos que los demás están poniendo sus problemas en nosotros y que están en contra nuestra, y eso está mal. Hay que ser comprensivos y discutir de qué forma pueden hacer funcionar mejor las cosas. Ese es el primer paso, entender cuáles son tus problemas y los de los demás. Luego de eso se puede empezar a alinear el negocio y mejorar los procesos. Irán notando pequeñas victorias. Es importante construir una cadena de esos logros, para saber lo que van a manejar o lo que va a empezar a suceder”.

La clave para lograr la integración de los equipos radica en comprender los problemas que puedan tener los compañeros de las otras áreas.

Recomendaciones iniciales

· Tener paciencia

Es importante entender que no se puede ni se tiene que cambiar todo desde el principio.

· Definir un proceso iterativo e incremental

Conviene empezar por objetivos pequeños y poco ambiciosos, para evaluar los resultados e ir mejorando.

· No hay que romper los procesos actuales de desarrollo

Las prácticas de DevOps pueden incorporarse en paralelo a los trabajos que se estén realizando dentro de la organización.

· No a los silos

Hay que evitar que los equipos trabajen de forma separada e independiente.

· Responsabilidad compartida

Se debe fomentar el sentido de responsabilidad grupal para mantener el producto y construir nuevos proyectos.

· Colaboración

Las reuniones son un buen método para integrar y alinear a las personas, pues permiten conocer en qué actividades están trabajando cada uno; facilitando la planificación y preparación de proyectos.

· Empatía

Las opiniones distintas pueden generar desacuerdos y conflictos. La forma correcta de resolver estas situaciones es hablar entre las partes para entender los diferentes alegatos y evaluar si, entre los distintos equipos, pueden llegar a una solución. Al final se trata de tener empatía y trabajar en conjunto.

· Pensar en el sistema

Es importante tener la visión global de todo lo que sucede. No se trata de que cada quien arregle la parte que le corresponde. Los sistemas son complejos y deben verse de forma global.

¿Cómo comenzar?

A las empresas puede resultarles difícil la adopción de la cultura Devops. Para tener una idea más clara de cómo comenzar, les compartimos esta metodología básica de implementación de proyectos DevOps:

Etapa 1 | Análisis general

Es importante entender que no se puede ni se tiene que cambiar todo desde el principio.

- ☑ Evaluar cuál es el **modelo de madurez** que tiene la organización.
- ☑ Crear una propuesta de ajustes de proceso, donde se incluyan las métricas que se van a medir.
- ☑ Elaborar un plan de capacitación para los equipos.
- ☑ Revisar con qué herramientas cuenta la organización y cuáles son sus alcances y limitaciones, pues el objetivo no es necesariamente la incorporación de nuevas tecnologías, sino aprovechar al máximo los recursos que se tienen.

Etapa 2 | Plan de trabajo

- ☑ Armar un equipo de trabajo, integrado por personas de distintas áreas. Este grupo comenzará el proceso de evangelización y de cambios de paradigmas dentro de la organización.
- ☑ Crear un Producto Mínimo Viable (PMV) y un plan de acciones. En este punto se deben definir las primeras prácticas que se van a implementar, ya sea en un proyecto pequeño o en un área específica de desarrollo.
- ☑ Ejecutar capacitaciones y workshops, incluyendo las que no se detectaron en la Etapa 1, pero sí se vislumbraron en esta fase.
- ☑ Implementar todas las tareas necesarias para llevar a cabo el PMV.
- ☑ Hacer feedback con el personal involucrado.

Etapa 3 | Revisión de los resultados y planificación del próximo proyecto

- ☑ Analizar los resultados de acuerdo con el alcance planteado.
- ☑ Hacer feedback sobre la adopción de esas prácticas y los resultados obtenidos.
- ☑ Planificar la próxima práctica que se desea incorporar.

¡Manos a la obra!

1. Equipos

Es común que en muchas empresas los equipos trabajen de forma separada. Por eso la unificación es uno de los cambios más importantes.

"La unificación de equipos puede lograrse ya sea que trabajen en el mismo lugar físico o no. En ambos casos deben estar disponibles para trabajar conectados y en forma conjunta, como tribu", explica Silvia Keymetlian, Technical Support Manager de GeneXus.

El diccionario de la Real Academia Española define como tribus a los grupos cuyos miembros tienen costumbres en común.

En la industria tecnológica, las tribus son equipos organizados por 5 y hasta 100 personas (idealmente no debería ser un número mayor a 100), que trabajan de forma conectada por un interés en común.

Tips para organizar equipos:

Pensar en grande. Un equipo DevOps no se construye solo con la gente de desarrollo y operaciones, sino también con los de testing, diseño, seguridad informática, entre otros. De esas fusiones surgen términos como:

- **DesignOps**

Integración del equipo de diseño con el equipo de desarrollo.

- **DevSecOps**

Integración del equipo de seguridad informática, desarrollo y operaciones.

Testing. Es ideal hacer tests en todas las etapas del proyecto. No hay que olvidar que algunas metodologías incluso se basan en test-driven development, para comenzar el desarrollo a partir de los tests.

Pair programming. Esta práctica, común y efectiva, consiste en realizar el desarrollo del sistema en pareja (tanto el código como las pruebas). Esto sirve también para hacer revisión de código.

2. Procesos

Arquitectura. Para determinar qué tipo de arquitectura utilizar, es necesario evaluar los pros y los contras de cada una de las opciones. Anteriormente mencionamos que los mercados requieren que las liberaciones de software se hagan de forma ágil. En estos casos, **trabajar con arquitecturas monolíticas no es lo más conveniente.**

"Cuando hay que hacer cambios en una funcionalidad o mover de lugar algún campo en una pantalla, hay que liberar el sistema entero y eso puede traer problemas. También puede suceder que otra parte de la aplicación no esté lista en el momento de querer liberar esos cambios, entonces hay una parte a medias y otra terminada, pero la persona que tiene su parte pronta tiene que esperar al resto del equipo, etc. Esto hace que el ciclo de liberación no sea para nada ágil, provoca caos, conflictos, etc. Por estas razones, la arquitectura de las aplicaciones está tendiendo a la modularización; para tratar de tener lo más desacoplado posible la arquitectura del sistema", explica Keymetlian.

Infraestructura. "Ahora lo más utilizado es la nube. Para gestionar la infraestructura, es aconsejable escribir un script con todos los requerimientos necesarios para armar el servidor y ejecutarlo cada vez que se deba crear ese ambiente. Esta es una forma de evitar errores humanos, garantizando que la configuración se haga de la misma manera. Una vez que tenemos definida la arquitectura de nuestro sistema, tenemos que ver cuáles son las cosas que debemos hacer todos los días para lograr un proceso ágil y automatizar todas las tareas posibles para que esto fluya y logremos esas entregas ágiles de las que estamos hablando".

Análisis. Una vez que la aplicación está en producción, se debe realizar un análisis para obtener aprendizajes y evaluar de qué forma se puede mejorar el código.

"Es importante entender cómo se comporta el código en producción. Tal vez se hicieron muchas pruebas sobre la aplicación, pero nunca se llegó a un pico de tráfico y recién cuando se lleva a producción es que se puede obtener esa información", agrega Silvia.

Métricas. Los resultados de las mediciones determinarán las vertientes que se deben ajustar en el plan. Algunas de las variables que se pueden medir son:

- ☑ El plazo de la entrega de los cambios.
- ☑ La frecuencia del deployment.
- ☑ El tiempo para restaurar el servicio en caso de fallas. En este punto se debe actuar con rapidez para eliminar los problemas y liberar la corrección, la cual debe seguir todos los pasos del pipeline.
- ☑ Tasa de fallas.
- ☑ Satisfacción laboral.

Test. Deben realizarse tests durante todo el proceso. La automatización de estas pruebas contribuirá a entregar productos de forma continua, con velocidad y calidad. De haber fallas, entonces se tendrá esa información antes de llevar a producción la aplicación.

3. Automatización

Una vez que se tengan definidos los procesos, se deben automatizar todas las tareas posibles para que todo fluya y se logren las entregas ágiles.

La automatización cede a las computadoras las tareas repetitivas que no requieren de pensamiento humano, permitiéndole al equipo concentrarse en los problemas que surjan y en sus posibles soluciones.

Las automatizaciones ayudan a que no se olvide la ejecución de ningún paso.

La automatización cede a las computadoras las tareas repetitivas que no requieren de pensamiento humano, permitiéndole al equipo concentrarse en los problemas que surjan y en sus posibles soluciones.

Algunas de las tareas que deben automatizarse para poder definir un pipeline:

- ☑ Code
- ☑ Build
- ☑ Test
- ☑ Integrate
- ☑ Deploy
- ☑ Release

Integración Continua. Consiste en la automatización de la integración y validación continua.

Por ejemplo, el código debe probarse primero de forma individual; y luego, si funciona bien, puede integrarse en todo el sistema.

Este método disminuye la probabilidad de que surjan errores y minimiza los conflictos que pudieran surgir.

Entrega Continua. Es importante realizar la automatización desde la implementación hasta la puesta en producción, y el armado del release.

La entrega continua se logra automatizando las distintas etapas del proceso de desarrollo, consiguiendo así armar un paquete con todo lo necesario para llevar la aplicación a producción. En esta fase lo único que se hace de forma manual es copiar el paquete al servidor donde quedará la aplicación disponible, y eso se hace con un solo clic.

DevOps. A esta fase se llega cuando están incluidos en el proceso la operación y monitoreo de la aplicación en producción. Así se genera la información que retroalimenta el plan y las acciones a seguir.

DevOps en el mundo GeneXus

Muchas organizaciones como Microsoft, Amazon y Google han usado DevOps para tener más agilidad y mejorar la forma en que se liberan las aplicaciones. GeneXus también se ha sumado a esta movida, implementando cambios en las formas de trabajo y en sus liberaciones, con el objetivo de dar más valor a las soluciones creadas por la comunidad.

GeneXus ha ido incorporando DevOps en las distintas etapas como testeo, chequeos de código estático, tareas de deploy, monitoreo de la aplicación en producción, entre otros.

El DevOps cubre más partes del ciclo de desarrollo ampliado de la aplicación, y sirve tanto para equipos grandes de desarrollo, como para pequeños grupos de desarrolladores.

Bajo la premisa de automatizar todo lo automatizable, GeneXus dispone de una variedad de herramientas que permiten hacer construcciones, despliegues, chequeos de API, chequeos de performance, testeos unitarios, testeos de integración, y mucho más.

Al tener un único Repositorio de referencia, se minimizan los errores manuales, generando un óptimo nivel de trazabilidad que identifica desde la versión instalada hasta los arreglos y datos de los cambios realizados y de los cambios no deseados.

GeneXus ha ido incorporando DevOps en las distintas etapas como testeo, chequeos de código estático, tareas de deploy, monitoreo de la aplicación en producción, entre otros.

Recomendaciones básicas para hacer DevOps con GeneXus

Usar GeneXus™ Server, ya que es la herramienta de Gestión de Configuración de Software para trabajar con GeneXus, y también la que permite unificar todas las tareas de automatización asociadas a DevOps.

Asegurarse de que todo el equipo de DevOps cuente con conocimientos en procesos de prueba.

Automatizar las pruebas desde el comienzo del proceso de desarrollo.

Hacer pruebas funcionales.

Potenciar la agilidad, automatización, seguridad, pruebas y monitorización utilizando herramientas como GeneXus Server, GXtest, Módulos, Log API, Deployment Units, MS Build, Jenkins, y Docker Containers.

Herramientas de GeneXus para DevOps

Estas son las herramientas que ofrece GeneXus para automatizar todo el proceso de DevOps:

GeneXus™ Server

Es el producto que automatiza la integración de conocimiento, mejorando las capacidades de trabajo en equipo, sin sumar costos a la integración (permite integrar al proyecto a gerentes o al cliente para el que se está trabajando, para que puedan supervisar el desarrollo del sistema).

Esta herramienta brinda la posibilidad de crear y monitorear Pipelines de Continuous Integration que permiten tener todo nuestro proceso de Continuous Integration integrado y automatizado.

GeneXus™ Server almacena toda la información sobre los cambios (cuándo y por qué fueron realizados), logrando un registro histórico que sirve para entender la evolución de los objetos, o para identificar los cambios que pueden haber producido cierto error y solucionarlo rápidamente.

Módulos

Es lo primero que se encuentra cuando se construyen las Bases de Conocimiento (Knowledge Base, o KB por sus siglas en inglés). Su función es ayudar con la organización, para que los desarrolladores sepan en qué módulos deben trabajar. Más adelante esta funcionalidad les dará la flexibilidad para poder empaquetar esos módulos y distribuirlos, facilitando el cambio de la arquitectura, si es necesario, de un sistema monolítico hacia otras opciones.

Importación de diseños externos

Para escenarios donde se precisan experiencias totalmente personalizadas a una realidad muy concreta, hemos potenciado la funcionalidad de importar diseños externos desde Sketch y Figma. De esta manera los diseñadores pueden trabajar en su herramienta favorita y se disminuye la fricción para llegar del diseño a una implementación pixel perfect.

GXtest

Es el producto desarrollado para diseñar, automatizar y ejecutar pruebas funcionales en aplicaciones web y móviles desarrolladas con GeneXus. ¿Lo mejor? Los usuarios de GXtest no necesitan contar con conocimientos en programación, ya que es intuitiva y muy fácil de usar.

Contenedores

GeneXus ofrece la opción de realizar el deploy en Contenedores y en otros ambientes como IBM, Amazon, Google, Azure, SAP, Dockers y Kubernetes, ayudando a que el despliegue de la aplicación se realice de forma mucho más rápida y sin los problemas de instalación que conlleva cuando se tiene que instalar un servidor. Todo esto es programable y automatizable, por eso es fácil incluirlo en el proceso de DevOps.

Deployment Units

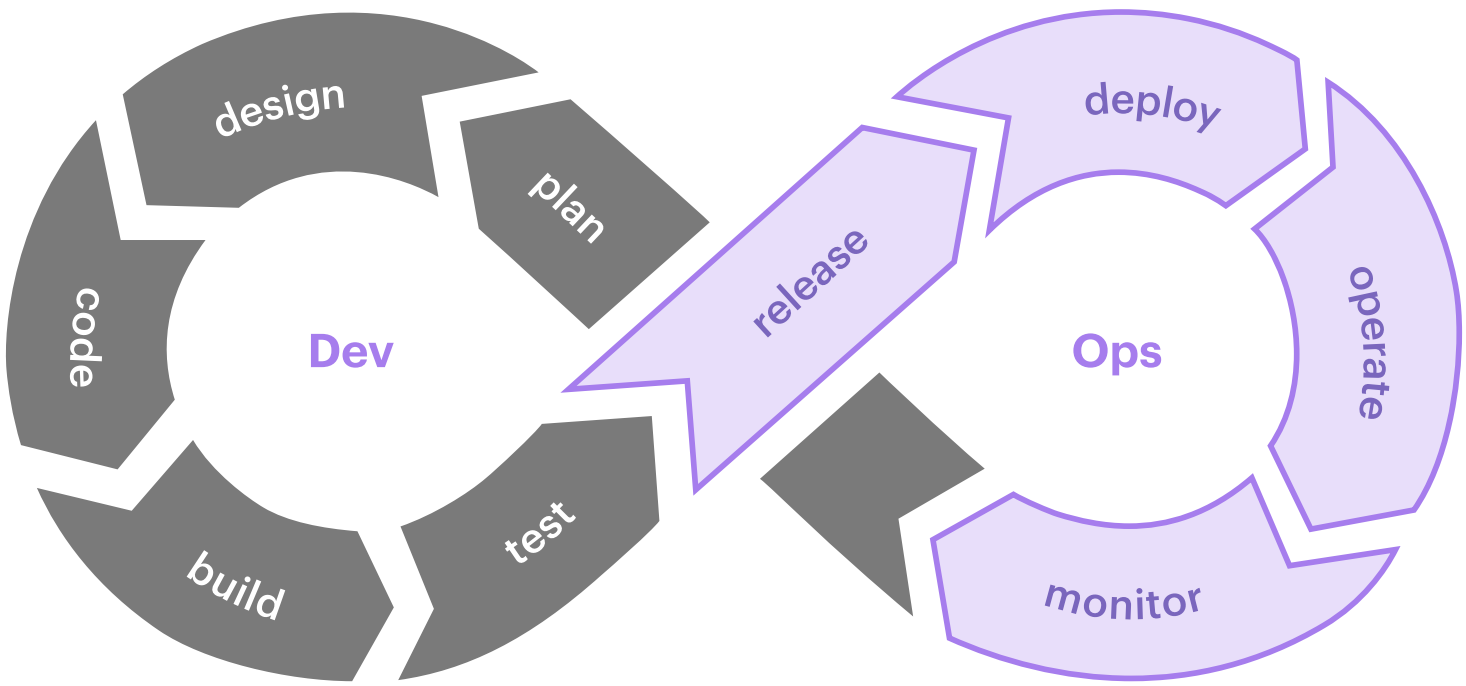
Es el despliegue que se hace a partir de la información que está en un objeto. Aunque se tenga una sola Knowledge Base (KB), esta funcionalidad facilita la creación de otras deployment units para cuando se necesite migrar de una Arquitectura Monolítica hacia una Arquitectura de Micro Servicios. Se pueden tener Deployment UNIT para los procesos batch, para los servicios móviles, para el frontend y para el backend, por ejemplo.

Log API

Permite al usuario realizar un monitoreo de la aplicación a través de la grabación (en los archivos de Log) sobre uno o varios procesos que se están ejecutando.
Con esto se obtiene información de cómo se comporta la aplicación y eso brinda información para retroalimentar el plan y que todo fluya en forma constante.

SSO GAM

Se usa para usar un solo login de usuario entre dos web app distintas, (de esta manera se evita tener que pedir el login nuevamente).



Descubre lo que GeneXus
puede hacer por tu empresa.

Contáctanos

info@genexus.com

GeneXus™

by Globant

MONTEVIDEO - URUGUAY	Av. Italia 6201- Edif. Los Pinos, P1	(598) 2601 2082
CIUDAD DE MÉXICO - MÉXICO	Hegel N° 221, Piso 2, Polanco V Secc.	(52) 55 5255 4733
MIAMI - USA	8950 SW 74th Ct, Suite 1406	(1) 201 603 2022
SÃO PAULO - BRASIL	Rua Samuel Morse 120 Conj. 141	(55) 11 4858 0300
TOKYO - JAPAN	2-27-3, Nishi-Gotanda	(81) 3 6303 9381
	Shinagawa-ku, Tokyo, 141-0031	(81) 3 6303 9980