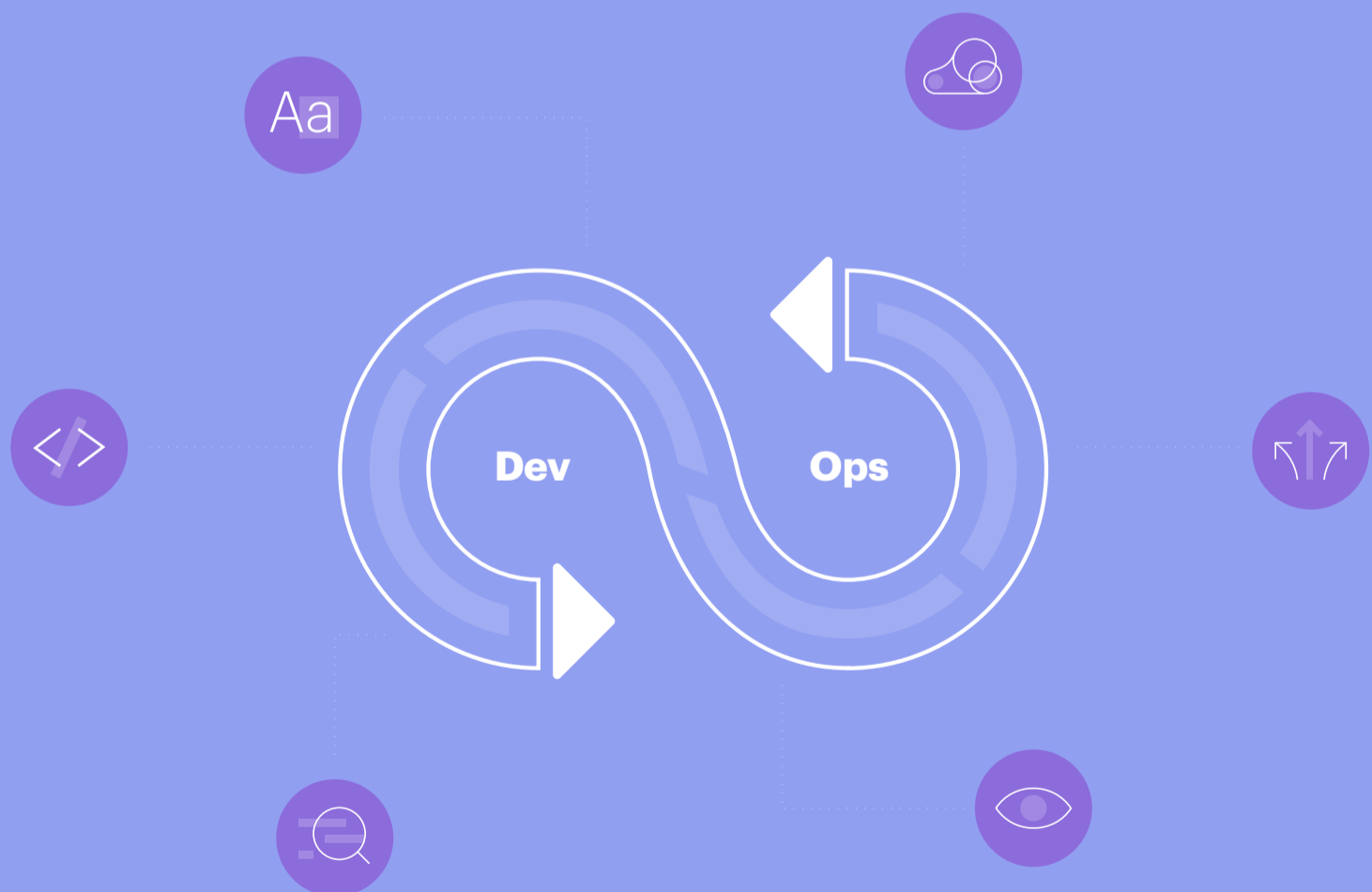


# DevOps: guia de implementação

Whitepaper



O mercado atual exige que o desenvolvimento e o lançamento de funcionalidades de software sejam cada vez mais ágeis.

Para cumprir essa entrega contínua de valor, é preciso avaliar e fazer mudanças nos processos, ferramentas e formas de trabalhar por meio de uma metodologia, própria da engenharia de software, conhecida como DevOps.

O termo, que é uma junção das palavras “desenvolvimento” e “operações”, foi inventado pelo consultor de tecnologia e programador [Patrick Debois](#) para resumir os benefícios de remover barreiras entre equipes de desenvolvimento e operações, por meio da implementação de ferramentas e processos ágeis.

A palavra ganhou relevância mundial em 2009 após o lançamento do [evento Devopsdays](#) (organizado pela Debois), que alcançou grande repercussão nas redes sociais e chamando a atenção de grandes empresas do setor como a IBM.

Abaixo detalhamos quais são as mudanças que organizações que desejam implementar uma cultura DevOps, e quais são os processos e ferramentas que devem ser utilizados para alcançar a entrega de um produto melhor.

## Por que começar no mundo DevOps?

As empresas de tecnologia sabem da importância de lançar seus produtos no mercado de forma ágil e contínua, mas é difícil para eles encontrar a melhor forma de resolver situações como:

- ☑ As barreiras que existem entre as diferentes equipes de trabalho.
- ☑ A fluidez entre colaborações internas.
- ☑ O aproveitamento dos recursos disponíveis.
- ☑ Agilidade na produção de aplicativos.
- ☑ Adoção da automação.

- ☑ A redução dos tempos de falha dos serviços.

- ☑ Segurança da informação.

DevOps is the working method for solving DevOps é a forma de trabalhar que ajuda a resolver muitos desses pontos.

Por meio de um conjunto de práticas, essa metodologia propõe uma nova forma de pensar e enfrentar as situações para reduzir as barreiras entre as pessoas e melhorar a automação dos processos em todas as etapas da cadeia de valor.

---

«O DevOps é focado em princípios, práticas e resultados. Para avançar para o DevOps, é muito importante apostar na preparação das equipes, compreender tudo o que envolve esta cultura, ajustar os processos ao que é realmente necessário e, por fim, instalar a tecnologia que ajude a implementação destas práticas»

Explica o consultor DevOps, [Genry Leiva](#) no webinar [Fazendo DevOps com GeneXus](#), oferecido com [Enrique Almeida](#), engenheiro de Software.

Em uma [entrevista realizada pela Devops-latam](#), Debois explicou que a chave para a integração da equipe está em entender os problemas que os colegas de outras áreas podem ter. “Às vezes pensamos que os outros estão colocando seus problemas em nós e que estão contra nós, e isso está errado. Seja compreensivo e discuta como você pode fazer as coisas funcionarem melhor. Esse é o primeiro passo, entender quais são os seus problemas e os dos outros. Depois disso, você pode começar a alinhar o negócio e melhorar os processos. Eles notarão pequenas vitórias. É importante construir uma cadeia dessas conquistas, saber com o que eles vão lidar ou o que vai começar a acontecer.”

A chave para a  
integração da  
equipe está em  
entender os  
problemas que os  
colegas de outras  
áreas podem ter.



## Recomendações iniciais

### • Ser paciente

É importante entender que você não pode e não precisa mudar tudo desde o início.

### • Definir um processo iterativo e incremental

É conveniente começar com objetivos pequenos e pouco ambiciosos, para avaliar os resultados e ir melhorando.

### • Não é necessário quebrar os atuais processos de desenvolvimento

As práticas de DevOps podem ser incorporadas paralelamente ao trabalho que está sendo realizado dentro da organização.

### • Não aos silos

Evite que as equipes trabalhem de forma separada e independente.

### • Responsabilidade compartilhada

Um senso de responsabilidade de grupo deve ser fomentado para manter o produto e construir novos projetos.

### • Colaboração

As reuniões são um bom método para integrar e alinhar as pessoas, pois permitem saber em quais atividades cada um está trabalhando; facilitando o planejamento e a preparação de projetos.

### • Empatia

As diferentes opiniões podem gerar divergências e conflitos. A maneira correta de resolver essas situações é conversar entre as partes para entender os diferentes argumentos e avaliar se, entre as diferentes equipes, elas podem chegar a uma solução. No final, trata-se de ter empatia e trabalhar em conjunto.

### • Pensar no sistema

É importante ter uma visão global de tudo o que acontece. Não se trata de todos fixarem apenas na parte que lhes corresponde. Os sistemas são complexos e devem ser vistos globalmente.

## Como começar?

As empresas podem achar difícil adotar a cultura Devops. Para ter uma ideia mais clara de como começar, compartilhamos esta metodologia básica para implementação de projetos DevOps:

### Etapa 1 | Análise geral

- ☑ Avaliar qual é o **modelo de maturidade** que a organização possui.
- ☑ Criar uma proposta de ajustes de processo, incluindo as métricas a serem medidas.
- ☑ Elaborar um plano de capacitação para as equipes.
- ☑ Revisar quais ferramentas a organização possui e quais são suas abrangências e limitações, pois o objetivo não é necessariamente a incorporação de novas tecnologias, mas sim aproveitar ao máximo os recursos disponíveis.

### Fase 2 | Plano de trabalho

- ☑ Montar uma equipe de trabalho, composta por pessoas de diferentes áreas. Este grupo iniciará o processo de evangelização e mudança de paradigma dentro da organização.
- ☑ Criar um Produto Mínimo Viável (MVP) e um plano de ação. Neste ponto, devem ser definidas as primeiras práticas a serem implementadas, seja em um pequeno projeto ou em uma área

específica de desenvolvimento.

- ☑ Realizar treinamentos e workshops, inclusive aqueles que não foram detectados na Fase 1, mas foram vislumbrados nesta fase.

- ☑ Implementar todas as tarefas necessárias para realizar o MVP.

- ☑ Dar feedback ao pessoal envolvido

### Fase 3 | Revisão dos resultados e planejamento do próximo projeto

- ☑ Analisar os resultados de acordo com o escopo proposto.

- ☑ Fornecer feedback sobre a adoção dessas práticas e os resultados obtidos.

- ☑ Planejar a próxima prática que você deseja incorporar.

# Mãos à obra!

## 1. Equipes

É comum que as equipes em muitas empresas trabalhem separadamente. É por isso que a unificação é uma das mudanças mais importantes.

“A unificação das equipes pode ser alcançada independentemente de trabalharem no mesmo local físico ou não. Em ambos os casos, eles devem estar disponíveis para trabalhar conectados e juntos, como uma tribo”, explica [Silvia Keymetlian](#), gerente de Suporte Técnico de GeneXus.

O dicionário da Real Academia Espanhola define tribos como grupos cujos membros têm costumes em comum.

Na indústria de tecnologia, tribos são equipes organizadas de 5 a 100 pessoas (idealmente não mais de 100), que trabalham de forma conectada por um interesse comum.

## Dicas para organizar equipes:

**Pensar grande.** Uma equipe de DevOps não é construída apenas com pessoas de desenvolvimento e operações, mas também com testes, design, segurança de TI, entre outros. Dessas fusões surgem novos termos como:

- ☑ **DesignOps**

Integração da equipe de design com a equipe de desenvolvimento.

- ☑ **DevSecOps**

Integração da equipe de segurança da informação, desenvolvimento e operações.

**Testes.** O ideal é fazer testes em todas as etapas do projeto. Não se deve esquecer que algumas metodologias são mesmo baseadas em desenvolvimento orientado a testes, para iniciar o desenvolvimento a partir de testes.

**Programação em pares.** Esta prática comum e eficaz consiste em realizar o desenvolvimento do sistema em pares (tanto o código como os testes). Isso também é útil para revisão de código.

## 2. Processos

**Arquitetura.** Para determinar qual tipo de arquitetura utilizar, é necessário avaliar os prós e contras de cada uma das opções.

Mencionamos anteriormente que os mercados exigem que os lançamentos de software sejam feitos rapidamente. Nesses casos, **trabalhar com arquiteturas monolíticas não é o mais conveniente.**

“Quando você precisa fazer alterações em um recurso ou mover um campo em uma tela, você precisa liberar todo o sistema e isso pode causar problemas. Também pode acontecer que outra parte do aplicativo não esteja pronta quando você quiser liberar essas alterações, então há uma parte semi-acabada e outra parte finalizada, mas a pessoa que tem sua parte pronta tem que esperar o resto da equipe, etc. Isso faz com que o ciclo de lançamento não seja nada ágil, causando caos, conflitos, etc. Por esses motivos, a arquitetura tende à modularização; tentar ter a arquitetura do sistema o mais desacoplada possível”, explica Keymetlian.

**Infraestrutura.** “Agora a mais utilizada é a nuvem. Para gerenciar a infraestrutura, é aconselhável escrever um script com todos os requisitos necessários para construir o servidor e executá-lo sempre que for necessário criar esse ambiente. Esta é uma forma de evitar erros humanos, garantindo que a configuração seja feita da mesma forma. Depois de definir a arquitetura do nosso sistema, temos que ver quais são as coisas que devemos fazer todos os dias para obter um processo ágil e automatizar todas as tarefas possíveis para que isso flua e alcancemos

as entregas ágeis de que estamos falando”.

**Análises.** Uma vez que o aplicativo está em produção, a análise deve ser realizada para obter aprendizados e avaliar como o código pode ser melhorado.

“É importante entender como o código se comporta em produção. Talvez muitos testes tenham sido feitos na aplicação, mas nunca se chegou a um pico de tráfego e só quando é levado para produção é que essa informação pode ser obtida”, completa Silvia.

**Métricas.** Os resultados das medições determinarão os aspectos que devem ser ajustados no plano. Algumas das variáveis que podem ser medidas são:

- O prazo de entrega das mudanças.
- A frequência da implantação.
- O tempo para restaurar o serviço em caso de falhas. Nesse ponto, você deve agir rapidamente para eliminar os problemas e liberar a correção, que deve seguir todas as etapas do pipeline.
- Taxa de falhas.
- Satisfação no trabalho.

**Teste.** Os testes devem ser realizados durante todo o processo. Automatizar esses testes ajudará a entregar produtos continuamente, com velocidade e qualidade. Se houver falhas, essas informações estarão disponíveis antes de levar o aplicativo para produção.



### 3. Automação

Uma vez definidos os processos, todas as tarefas possíveis devem ser automatizadas para que tudo flua e as entregas ágeis sejam alcançadas.

A automação transfere tarefas repetitivas que não exigem pensamento humano para os computadores, permitindo que a equipe se concentre em problemas emergentes e possíveis soluções.

As automações ajudam a garantir que nenhuma etapa seja esquecida.

A automação transfere tarefas repetitivas que não exigem pensamento humano para os computadores, permitindo que a equipe se concentre em problemas emergentes e possíveis soluções.

Estas são algumas das tarefas que devem ser automatizadas para definir um pipeline:

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

**Integração Contínua.** Consiste em automatizar a integração e validação contínua.

Por exemplo, o código deve primeiro ser testado individualmente; e então, se funcionar bem, pode ser integrado em todo o sistema.

Este método diminui a probabilidade de erros e minimiza os conflitos que podem surgir.

**Entrega Contínua.** É importante automatizar desde a implementação até a produção e liberar a montagem.

A entrega contínua é conseguida automatizando as diferentes etapas do processo de desenvolvimento, conseguindo assim montar um pacote com tudo o que é necessário para levar a aplicação à produção. Nesta fase, a única coisa que é feita manualmente é copiar o pacote para o servidor onde a aplicação estará disponível, e isso é feito com um único clique.

**DevOps.** Essa fase é alcançada quando a operação e o monitoramento do aplicativo em produção são incluídos no processo. Isso gera as informações que retroalimentam o plano e as ações a seguir.

## DevOps no mundo GeneXus

Muitas organizações como Microsoft, Amazon e Google têm utilizado DevOps para ser mais ágil e melhorar a forma como os aplicativos são lançados. GeneXus também aderiu a este movimento, implementando mudanças nas formas de trabalhar e em seus lançamentos, com o objetivo de dar mais valor às soluções criadas pela comunidade.

GeneXus vem incorporando DevOps em diferentes etapas como testes, verificações de código estático, tarefas de implantação, monitoramento da aplicação em produção, entre outras.

O DevOps abrange mais partes do ciclo de desenvolvimento de aplicativos estendido, atendendo a grandes equipes de desenvolvimento e pequenos grupos de desenvolvedores.

Sob a premissa de automatizar tudo o que pode ser automatizado, GeneXus possui uma variedade de ferramentas que permitem construções, implantações, verificações de APIs, verificações de desempenho, testes unitários, testes de integração e muito mais.

Por ter um único Repositório de referência, os erros manuais são minimizados, gerando um nível ótimo de rastreabilidade que identifica desde a versão instalada até as correções e dados das alterações realizadas e indesejadas.

GeneXus vem incorporando DevOps em diferentes etapas como testes, verificações de código estático, tarefas de implantação, monitoramento da aplicação em produção, entre outras.

## Recomendações básicas para fazer DevOps com GeneXus

Usar GeneXus™ Server, pois é a ferramenta de gerenciamento de configuração de software para trabalhar com GeneXus, e também aquela que permite unificar todas as tarefas de automação associadas ao DevOps.

Certificar-se de que toda a equipe de DevOps tenha conhecimento em processos de teste.

Automatizar testes desde o início do processo de desenvolvimento.

Realizar testes funcionais.

Potencializar a agilidade, automação, segurança, testes e monitoramento usando ferramentas como GeneXus Server, GXtest, Módulos, Log API, Deployment Units, MS Build, Jenkins e Docker Containers.

# Ferramentas GeneXus para DevOps

Estas são as ferramentas que GeneXus oferece para automatizar todo o processo DevOps:

## **GeneXus™ Server**

Este é o produto que automatiza a integração do conhecimento, melhorando a capacidade de trabalho em equipe, sem adicionar custos à integração (permite integrar a gerentes de projeto ou o cliente para o qual você está trabalhando, para que possam supervisionar o desenvolvimento do sistema).

Esta ferramenta oferece a possibilidade de criar e monitorar Continuous Integration Pipelines que permitem integrar e automatizar todo o nosso processo de Integração Contínua.

GeneXus™ Server armazena todas as informações sobre as mudanças (quando e por que foram feitas), conseguindo um registro histórico que é útil para entender a evolução dos objetos, ou para identificar as alterações que podem ter causado um determinado erro e resolvê-lo rapidamente.

## **Módulos**

É a primeira coisa que se encontra ao construir as Bases de Conhecimento (Knowledge Base, ou KB, por sua sigla em inglês). Seu papel é ajudar a organização, para que os desenvolvedores saibam em quais módulos precisam trabalhar. Mais tarde, essa funcionalidade lhes dará a flexibilidade de

poder empacotar esses módulos e distribuí-los, facilitando a mudança da arquitetura, se necessário, de um sistema monolítico para outras opções.

## **Importação de projetos externos**

Para cenários onde são necessárias experiências totalmente customizadas para uma realidade muito específica, aprimoramos a funcionalidade de importação de projetos externos do Sketch e Figma. Dessa forma, os designers podem trabalhar em sua ferramenta favorita e o atrito para passar do design para uma implementação perfeita de pixel é reduzido.

## **GXtest**

É o produto desenvolvido para desenhar, automatizar e executar testes funcionais em aplicações web e mobile desenvolvidas com GeneXus. O melhor? Os usuários do GXtest não precisam de conhecimentos de programação, pois é intuitivo e muito fácil de usar.

## **Containers**

GeneXus oferece a opção de implantar em Containers e em outros ambientes como IBM, Amazon, Google, Azure, SAP, Dockers e Kubernetes, ajudando a aplicação a ser implantada muito mais rápido e sem os problemas de instalação que implica quando você tem que instalar um servidor. Tudo isso é programável e automatizável, por isso é fácil incluí-lo no processo de DevOps.

## Unidades de Implantação

É a implantação que é feita a partir das informações que estão em um objeto. Embora exista apenas uma [Base de Conhecimento](#) (KB), essa funcionalidade facilita a criação de outras unidades de implantação para quando for necessário migrar de uma Arquitetura Monolítica para uma Arquitetura de Microsserviços. Você pode ter uma Deployment UNIT para processos em lote, para serviços móveis, para front-end e back-end, por exemplo.

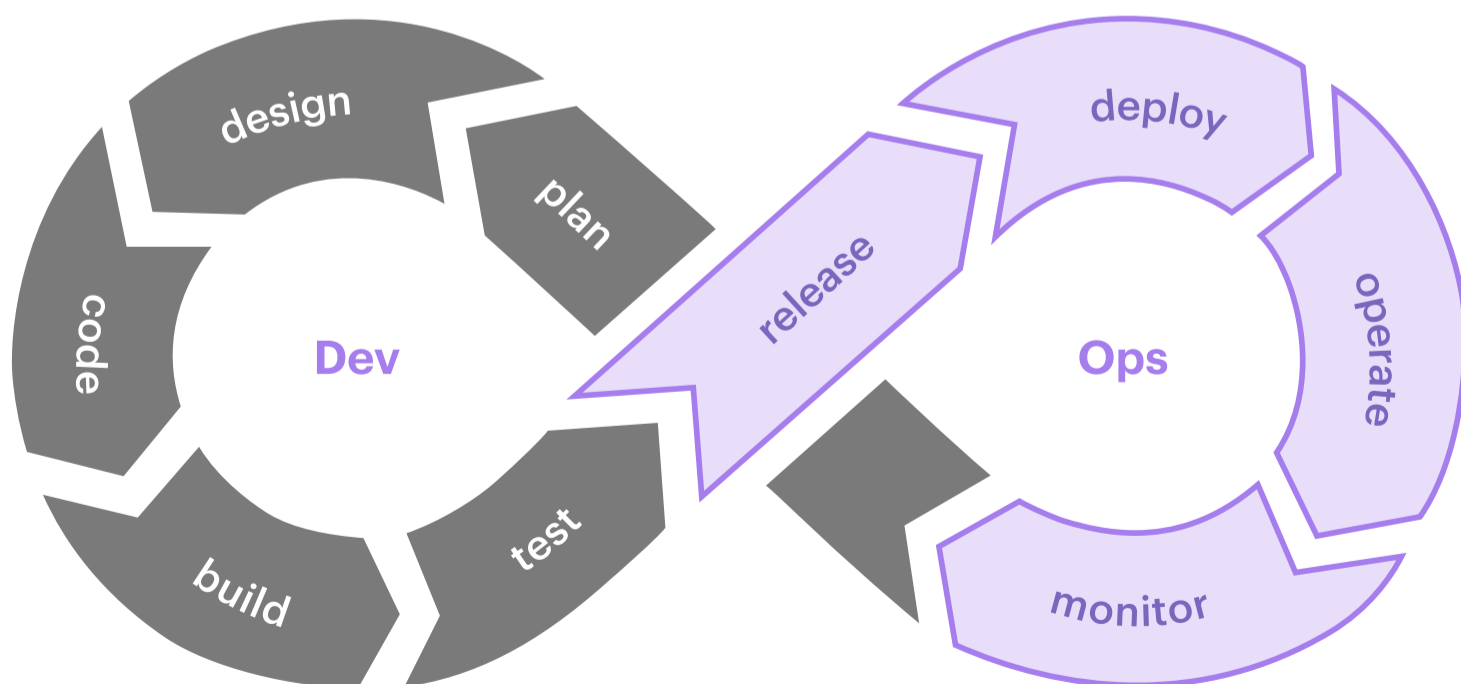
## Log API

Permite que o usuário monitore a aplicação registrando (nos arquivos de Log) um ou mais processos em execução.

Com isso, é possível obter a informação de como a aplicação se comporta e fornece informações para retroalimentar o plano e tudo flui de forma constante.

## SSO GAM

É usado para usar um único login de usuário entre dois aplicativos web diferentes, (assim você evita ter que solicitar o login novamente).



Descubra como GeneXus pode fazer o mesmo pela sua empresa.

[info@genexus.com](mailto:info@genexus.com)



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