

A blurred background image showing two people from the chest up. One person on the left is wearing a light grey shirt and a dark tie, looking at a silver tablet. Another person on the right is wearing a blue and white striped shirt, also looking at a silver tablet. A third person's hands are visible in the foreground, holding a silver smartphone.

Arquitetura e estratégia de API: uma abordagem coordenada

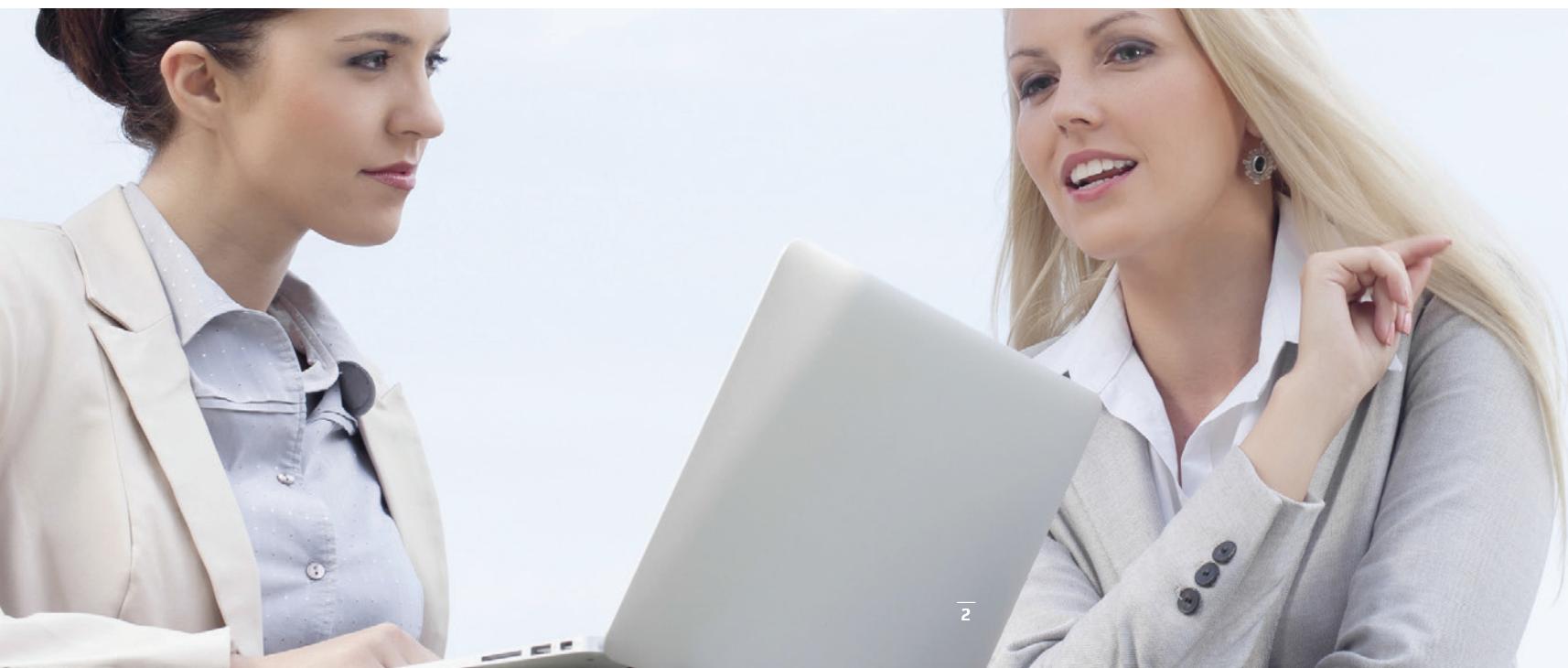
Introdução

A ascensão da API (Application Programming Interface - Interface de Programação de Aplicativos) representa uma oportunidade de negócios e um desafio técnico. Para os líderes de negócios, as APIs oferecem a oportunidade de gerar novos fluxos de receita e maximizar o valor para o cliente. Os arquitetos corporativos, por sua vez, são os responsáveis pela criação de APIs que disponibilizam sistemas de back-end para reutilização em novos aplicativos web e móveis.

É essencial que todas as partes interessadas compreendam que os objetivos de negócios e desafios técnicos de um programa de API estão intimamente relacionados. Os gerentes de programa devem assumir a responsabilidade de comunicar claramente os principais objetivos de negócios de uma API proposta aos arquitetos responsáveis pela criação da interface.

Os arquitetos, nesse caso, devem assumir a responsabilidade de manter um foco claro nesses objetivos ao longo do processo de implantação de uma infraestrutura de API e de criação da interface. Todas as decisões técnicas deverão contribuir para a criação de uma interface que permita aos desenvolvedores criar aplicativos cliente que possam gerar um valor real para os usuários finais.

Este eBook descreve as melhores práticas para a criação de APIs focadas em resultados que serão a base para o sucesso do seu programa de API.



Parte 1: de SOA para API

A TI corporativa do século XXI é caracterizada por uma fase de transição para a abertura de bancos de dados e aplicativos que não eram integrados, para que os dados e a funcionalidade possam ser acessados além das fronteiras organizacionais ou reutilizados em novos sistemas. A manifestação inicial dessa tendência foi a SOA (Service Oriented Architecture - Arquitetura Orientada a Serviços) e a mais recente foi o aumento repentino das APIs orientadas à web.

Por um lado, os "serviços web" essenciais para a SOA representam o mesmo que as APIs web. Ambos são interfaces usadas para abrir sistemas de back-end. No entanto, existem algumas diferenças fundamentais entre as duas tecnologias que são muito relevantes para as decisões básicas de criação:

- A principal diferença técnica é que os programas de SOA se concentram na criação de serviços web para possibilitar integrações internas, de servidor para servidor, ao passo que as APIs web existem para acelerar a criação de aplicativos web e móveis, frequentemente orientados a clientes.
- Os programas de SOA geralmente são conduzidos por departamentos de TI e concentrados na redução de custos, mas os programas de API costumam surgir com organizações de desenvolvimento de negócios e se concentrar na geração de novas receitas.
- A maioria dos projetos de SOA é criada para e pelos arquitetos corporativos, a fim de ajudá-los a integrar mais facilmente sistemas heterogêneos e oferecer novos serviços de TI. Os programas de API, pelo contrário, devem se concentrar em atender às necessidades dos desenvolvedores de aplicativos.

Figura 1: SOA versus APIs

	 SOA	 APIs
 Objetivo de integração	Interno ou para parceiros	Externo, frequentemente para clientes
 Direcionador do projeto	Custos de TI	Receitas de negócios
 Consumidor da interface	Arquitetos corporativos	Desenvolvedores de aplicativos

Objetivos da criação de APIs

Apesar de tudo, muitos programas de API surgem a partir de iniciativas anteriores de SOA. Serviços web concentrados em integrações internas ou com parceiros estão sendo abertos para os desenvolvedores — dentro e fora da empresa. Durante esse processo, é importante que os criadores de APIs se lembrem de que um programa de API tem drivers e requisitos muito diferentes daqueles que inicialmente incentivaram as empresas a abrirem seus ativos de TI por meio de serviços web.

Com isso em mente, os objetivos gerais da criação de APIs podem ser definidos como:

- Habilitar o autoatendimento para desenvolvedores e usuários de aplicativos.
- Reduzir as barreiras de acesso a recursos corporativos importantes.
- Priorizar as necessidades e as preferências dos desenvolvedores de aplicativos cliente.
- Incentivar a colaboração entre recursos internos e externos.
- Lidar com os problemas de segurança e escalabilidade gerados após a exposição dos ativos de TI para o mercado aberto.

Acima de tudo, a criação de APIs deve se concentrar em maximizar o valor comercial da interface. Na segunda parte, observaremos mais de perto como as APIs agregam valor aos negócios.



Parte 2: a cadeia de valor da API

As APIs podem não ter um valor intrínseco, mas geram um grande valor para os negócios. Elas fazem isso por meio de seus dados de back-end e da funcionalidade dos aplicativos que a interface habilita. Nesse caso, a API é simplesmente um facilitador que permite que sistemas com grande valor organizacional sejam reutilizados em aplicativos mais propensos a gerar um valor comercial direto.

Embora essa seja uma perspectiva útil, quando observamos mais de perto, fica claro que uma API bem projetada é, na verdade, um conector complexo e poderoso. Ela consegue unir uma grande variedade de ativos de negócios — sistemas de TI, pessoal interno e externo, clientes e aplicativos cliente — para aproveitar o valor potencial desses ativos

com mais eficiência. Podemos chamar essa ocorrência de "cadeia de valor da API".

É importante compreender que uma API gera valor dessa maneira relativamente complexa, pois é muito fácil se esquecer do fato de que as APIs existem para gerar valor comercial, em vez de eficiência técnica. Embora as APIs gerem valor mais diretamente do que a SOA, elas trabalham de uma maneira menos direta do que a web com base em navegadores — na qual um site pode gerar vendas potenciais ou reais. As APIs geram receitas de uma forma mais sutil, vinculando os vários ativos descritos abaixo.

Figura 2: a cadeia de valor da API



Alguns exemplos de como as APIs geram valor

Cada API terá um valor exclusivo. Em linhas gerais, porém, as empresas podem usar uma API como uma forma de:

Gerar novas receitas diretamente

Uma API pode ser uma fonte direta de receita se os desenvolvedores forem cobrados pelo acesso ou se a interface for usada para possibilitar a criação interna de aplicativos pagos ou para permitir o comércio eletrônico.

Ampliar a obtenção de clientes e o valor agregado

As APIs simplificam o processo de obtenção de novos clientes ou de aumento do valor para os clientes existentes, oferecendo os serviços existentes por meio de novas plataformas e dispositivos.

Oferecer suporte a atividades de vendas e marketing

Uma API também ajuda as empresas a comercializarem seus produtos e serviços, permitindo a criação do tipo ideal de funcionalidade envolvente e imersiva associada às melhores práticas de marketing online.

Promover os negócios e a inovação técnica

As APIs ajudam as organizações a desenvolver novos sistemas, ofertas e estratégias porque reduzem as barreiras para a inovação, possibilitando a implementação de ideias sem alterar os sistemas de back-end.

Tomando decisões de criação

As decisões de criação de APIs devem ser direcionadas precisamente pelos ativos que a API vinculará — o que estará em cada lado da interface, tanto dentro da infraestrutura de TI organizacional quanto fora do firewall corporativo. Especificamente, é essencial responder a estas duas perguntas:

- Quais sistemas estão sendo expostos e onde (e com quem) eles residem?
- Quem é o público de desenvolvedores e que tipo de aplicativo será criado por eles?

"Quem é o público de desenvolvedores?" é uma pergunta particularmente importante e relevante para a categorização mais fundamental das APIs — como "privadas" ou "abertas". As APIs privadas são apenas para uso interno da empresa ou, em alguns casos, de organizações parceiras. As APIs abertas são disponibilizadas para a comunidade de desenvolvedores externos, que são livres para criar seus próprios aplicativos usando os recursos de back-end da empresa.

A natureza das APIs privadas é mais parecida com a dos serviços web. Geralmente, o objetivo de uma API privada será ajudar os desenvolvedores internos, prestadores de serviços ou parceiros a criar com mais eficiência aplicativos para uso interno ou externo. Assim como acontece nos serviços web, a redução de custos geralmente representa o principal direcionador, pois as APIs permitem que novos aplicativos sejam desenvolvidos de forma econômica. No entanto, muitas APIs privadas são usadas para criar aplicativos web e móveis voltados para o público que geram um novo valor comercial mais diretamente.

Os programas de API aberta costumam se concentrar na adoção. Ao permitir que os desenvolvedores de terceiros accessem suas APIs, as empresas visam disponibilizar seus ativos de TI para o maior número possível de usuários.

Portanto, a adoção dos desenvolvedores é uma métrica essencial para avaliar o sucesso de uma API aberta. Embora exista um número menor de APIs abertas do que de APIs privadas, as maiores oportunidades de negócios e os desafios de criação/riscos técnicos mais significativos são encontrados nas APIs abertas.

Na verdade, além de criar uma gama de desafios de criação de integração completamente novos (por exemplo, como abrir sistemas de back-end para desenvolvedores externos sem expor esses sistemas para hackers), as APIs abertas também geram novos riscos para os negócios. Um programa de API aberta criado de forma inadequada pode fazer com que uma empresa canibalize seu negócio principal e corra o risco de expor seus ativos de negócios essenciais para os concorrentes.

Considerações de negócios como essas devem direcionar as decisões relacionadas à criação técnica. Discutiremos como alinhar as considerações de negócios com as decisões técnicas na parte três.



Parte 3: alinhando a criação de APIs com os objetivos de negócios

Ao longo dos anos, embora a SOA procure melhorar os processos organizacionais, o objetivo dos programas de API é aumentar as receitas de negócios. Portanto, as decisões de criação de APIs devem se concentrar claramente nos principais objetivos estratégicos de negócios do programa de API da empresa. Antes de começar a criar uma API, você deve entender com clareza quais problemas o programa de API visa resolver, quais oportunidades devem ser aproveitadas e o caminho a ser seguido para atingir esses objetivos. Especificamente, é importante responder a estas perguntas:

- Quais ativos serão disponibilizados?
- Como a API deve disponibilizar esses ativos?
- Que tipo de aplicativo pode ser criado com base na API?
- Como os desenvolvedores podem ser motivados a usar a API?
- Como os aplicativos gerarão valor para a empresa?

A comunicação e a colaboração são fundamentais para a criação de uma API adequada para enfrentar esses desafios e aproveitar essas oportunidades. Durante o processo de criação, implantação e gerenciamento de uma interface, os gerentes de programa e os arquitetos de API devem trabalhar em conjunto para garantir que todos concordam com os principais objetivos estratégicos, com o que será feito para atingir esses objetivos e com a maneira de avaliar os resultados de seus esforços. Especificamente, as funções técnicas e de negócios devem entrar em acordo sobre:

- O objetivo e o estado final ideal do programa.
- As tarefas iniciais que permitirão à organização trabalhar para atingir esses objetivos.
- As principais métricas que serão usadas para avaliar o sucesso.
- As tarefas diárias em andamento que permitirão que o programa continue atingindo suas metas.

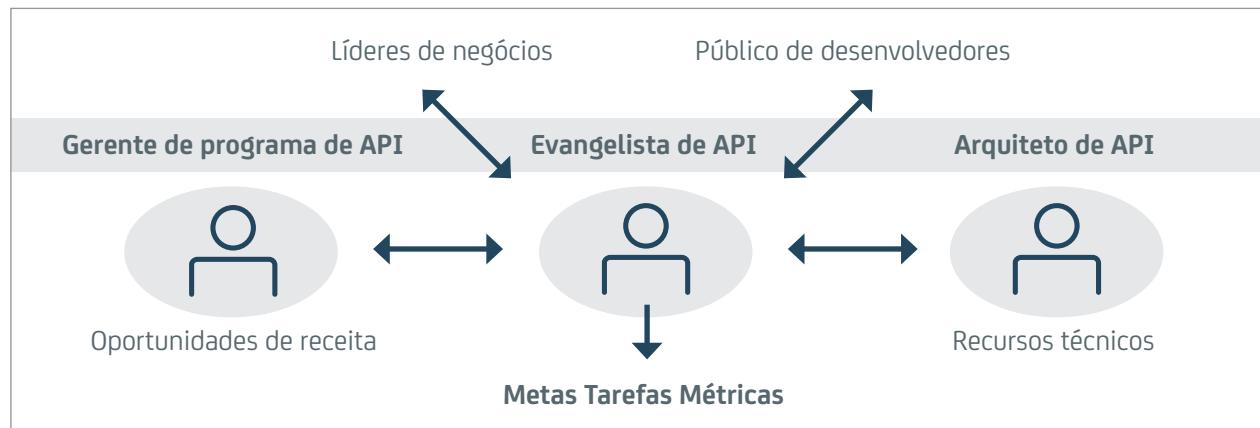
Atribuindo um patrocinador

Para garantir a sinergia entre os gerentes de negócios e os arquitetos, o programa deve ter um "patrocinador", capaz de preencher a lacuna que muitas vezes surge entre os departamentos técnicos, os gerentes de negócios e os desenvolvedores de aplicativos. Frequentemente, as organizações cometem o erro de atribuir essa função a um gerente de marketing sem experiência técnica, mas esse "evangelista de API" deve ser capaz de compreender as restrições da arquitetura da organização e compartilhar o entusiasmo dos desenvolvedores de aplicativos.

A função do evangelista é estabelecer uma comunicação clara com todas as partes interessadas, especificamente:

- "Vendendo" o programa de API para executivos e outros tomadores de decisões seniores.
- Garantindo que os arquitetos de API compreendam os objetivos de negócios dos gerentes.
- Ajudando os gerentes de programa a compreender os recursos técnicos e as restrições dos arquitetos.
- Reunindo informações sobre as preferências e os requisitos do público de desenvolvedores.

Figura 3: alinhando os objetivos da API



Uma vez que a comunicação tiver sido estabelecida, e as métricas, as tarefas e os objetivos tiverem sido acordados, o verdadeiro trabalho de criação de APIs poderá ser iniciado, um processo que será discutido na parte quatro.

Algumas observações sobre a estratégia comercial da API

Os gerentes de programa (ou "proprietários da API") — em colaboração com o evangelista de API da organização — devem assumir a responsabilidade pela elaboração de uma estratégia comercial clara de API e pela comunicação dessa estratégia aos tomadores de decisões de nível executivo, bem como aos arquitetos e desenvolvedores que implementarão a parte técnica da estratégia.

A primeira etapa é estabelecer um objetivo de negócios claro e elaborar a visão do programa de API com base na visão geral da empresa. Uma API não é uma solução puramente técnica, por isso, deve ser tratada como um produto ou uma estratégia comercial — embora esteja incorporada à estratégia comercial geral da empresa.

Com isso em mente, a próxima etapa deve ser criar um modelo de negócios com base nessa visão, definindo detalhes de:

Custos, recursos e eficiências

- Sistemas, relacionamentos, atividades e outros recursos que o programa utilizará e a maneira como o programa capacitará a empresa para que ela utilize melhor esses recursos.

Valor, receita e inovação

- Clientes, mercados e canais que serão visados pelo programa e a maneira como a inovação técnica possibilitará a geração de novas receitas a partir dessas metas.

A essência desse modelo de negócios deve ser uma proposta de valor que descreve claramente o valor comercial real e mensurável que o programa de API oferecerá para os negócios.

Parte 4: criando uma API utilizável

Do ponto de vista técnico, criar uma API é uma tarefa relativamente fácil. No entanto, a criação de uma API que possa gerar um valor real para os negócios é algo um pouco mais complexo. Além da funcionalidade, os arquitetos corporativos também devem considerar os objetivos de negócios e a experiência do usuário final.

Essa pode ser uma tarefa particularmente desafiadora para quem estiver trabalhando na extensão de um projeto de SOA para o grupo de APIs. Em um projeto de SOA, as necessidades do arquiteto são essenciais e a adoção do usuário é prevista. Consequentemente, os arquitetos com experiência em SOA costumam lidar com as decisões de criação de APIs com base na premissa de que os usuários da interface e dos aplicativos terão as mesmas necessidades e tendências que eles tiverem. O resultado disso, com frequência, são decisões inadequadas de criação.

Com as APIs, o foco da criação não deve ser a funcionalidade, mas a experiência do usuário. A questão fundamental não é "Que funcionalidade preciso expor?", mas "Como os desenvolvedores usarão essa interface?". Se os desenvolvedores não quiserem usar sua API, ela não terá valor. Portanto, a criação deve ser centrada nos desenvolvedores e focada em reduzir ao máximo a barreira de entrada para esse público de desenvolvedores.

Independentemente de se tratar de uma API privada ou aberta, uma boa DX (Developer Experience - Experiência de Desenvolvedor) será essencial para seu sucesso. Uma DX é significativamente mais difícil de quantificar do que a funcionalidade exposta. Embora ela possa ser definida como a soma das interações entre o fornecedor de API e o desenvolvedor, o resultado dessa soma é considerado mais uma percepção do que um número: como os desenvolvedores se sentem com relação à interface?

Obviamente, essa é uma métrica bastante indistinta, mas existem medidas práticas que podem ser aplicadas no mundo real para entender como é provável que seus desenvolvedores se sintam com relação às diferentes abordagens para a criação da sua API. Especificamente, você deve:

- Criar perfis de desenvolvedores.
- Gerar um protótipo e testar sua API em campo.

Perfis de desenvolvedores

Você só poderá criar uma API utilizável se conhecer as necessidades e as preferências do público de desenvolvedores. Há uma tendência em presumir que os desenvolvedores que criam aplicativos cliente com base em APIs são jovens que se consideram "hackers", obcecados pelos mais recentes protocolos e linguagens. Em muitos casos, porém — principalmente em cenários de APIs privadas —, os desenvolvedores de serviços corporativos continuam leais às maneiras mais consagradas de agir.

A questão é que cada projeto de API deverá lidar com um público específico de desenvolvedores para ter sucesso. Em alguns casos, o grupo pode ser muito homogêneo, com necessidades compartilhadas. Em outros, talvez seja necessário lidar com uma grande variedade de preferências. Independentemente disso, você deve compreender quem usará sua API e como você pode definir a interface para garantir que esses desenvolvedores possam usar seus recursos de back-end de maneira rápida e eficaz.

A primeira etapa, portanto, é elaborar uma persona (ou um conjunto de personas) para definir o tipo (ou tipos) de desenvolvedor que será o público de suas APIs. Essa etapa deve incluir informações sobre:

- Para quem os desenvolvedores trabalham (e em que departamento) e por que eles estão desenvolvendo um aplicativo.
- Habilidades de programação, restrições técnicas e preferências de linguagem/protocolo.
- Temperamento pessoal e em que contexto eles trabalham melhor.

Prototipagem

Após compreender os objetivos do trabalho, os requisitos técnicos e as preferências pessoais do seu público de desenvolvedores, você poderá iniciar a criação de uma interface que aborda esses critérios. No entanto, antes de criar uma API de produção vinculada a dados reais ou a sistemas de back-end, você deverá criar um protótipo leve que possa ser alterado com mais facilidade. Esse protótipo permitirá que você teste as premissas de criação que foram feitas com base na persona de destino.

Figura 4: ferramentas úteis de prototipagem de APIs

	1	Apiary apiary.io	Uma ferramenta de criação que permite gerar um protótipo de API com rapidez, sem a necessidade de escrever códigos.
Existem várias ferramentas online que simplificam o processo de criação e teste de protótipos leves de API. Veja alguns exemplos de ferramentas populares:	2	RAML raml.org	Linguagens de descrição de API que ajudam os desenvolvedores a encontrar e começar a usar seu protótipo de interface.
	3	SWAGGER swagger.io	

Uma das vantagens da criação de um protótipo leve com base em dados ou funcionalidades "descartáveis" é a possibilidade de aplicar uma segurança mínima e reduzir ao máximo a barreira de entrada para os desenvolvedores. Com isso, é possível conquistar seu público de desenvolvedores desde o início. Eles poderão criar aplicativos leves para testar sua API e dar feedback. Em seguida, você poderá fazer alterações na interface e testá-la novamente. Após algumas iterações, você estará no caminho certo.

É claro que nenhuma dessas opções aborda a maneira como você tomará decisões fundamentais e reais sobre a criação da interface. Na parte cinco, começaremos a discutir as opções reais de criação de APIs.

Parte 5: estilos de API

A escolha de um estilo de API é uma das decisões mais importantes que um criador de interfaces precisa tomar. Decisões desse tipo serão inevitavelmente influenciadas por considerações técnicas, como a natureza específica dos recursos de back-end que serão expostos ou as restrições da organização de TI. No entanto, outros aspectos também devem ser considerados, como os objetivos de negócios do programa de API, além das necessidades e preferências do público de desenvolvedores.

Atualmente, os estilos comuns de criação de APIs podem ser categorizados como:

Serviço web
(também conhecido
como Tunneling)

REST pragmático
(também conhecido
como URI)

Hipermídia (também
conhecido como
"True REST")

Orientado a eventos
(também conhecido
como IoT)

Serviço web

O estilo de serviço web é uma abordagem indiferente quanto ao transporte e com base em operações para a criação de APIs, que usa a WSDL (Web Services Description Language - Linguagem de Descrição de Serviços Web) para descrever as interfaces. Ele surgiu no mundo da SOA, no qual as interfaces de serviço web eram usadas para integrar redes heterogêneas. Portanto, essa pode ser uma boa opção de estilo se seu programa envolver a extensão de interfaces de SOA. A grande quantidade de ferramentas que existe para serviços web também significa que os aplicativos cliente geralmente podem ser criados de forma rápida e fácil.

A escolha desse estilo, porém, gera grandes limitações. Em primeiro lugar, embora esse estilo indiferente quanto ao transporte possa usar HTTP (Hypertext Transfer Protocol - Protocolo de Transferência de Hipertexto), ele é muito ineficiente neste contexto. Portanto, essa não é a melhor opção se seus serviços fizerem parte de uma extensão para a web aberta. Além disso,

REST pragmático

O estilo de REST (Representational State Transfer - Transferência de Estado Representacional) pragmático é uma abordagem mais simples e centrada na web para a criação de interfaces de integração. Esse estilo, que usa URI em vez de WSDL e é específico quanto ao transporte (oferece suporte exclusivamente para HTTP), substituiu o estilo de serviço web em grande parte da criação corporativa de APIs. O termo "API da web" geralmente é usado como sinônimo de "API RESTful", e atingir "RESTfulness" costuma ser considerado um objetivo importante de qualquer projeto de criação de interface.

Na verdade, a maioria das APIs REST em uso na atualidade não atende a todos os critérios de REST descritos na tese de doutorado de Roy Fielding, de 2000. Embora o REST tenha sido definido para descrever formalmente os tipos de interação dinâmica e com hiperlinks que dominam a web, a maioria das APIs da web lida com a troca de dados estáticos. Portanto, a título meramente argumentativo, é mais preciso se referir a este estilo de criação como "REST pragmático".

esse estilo só é prático se seu público de desenvolvedores estiver familiarizado com os padrões de SOA, como WSDL, SOAP (Simple Object Access Protocol - Protocolo Simples de Acesso a Objetos) e RPC (Remote Procedure Call - Chamada de Procedimento Remoto). Para a maioria dos desenvolvedores cliente, é provável que a curva de aprendizado seja íngreme.

Isso acontece com frequência em cenários de API aberta, principalmente naqueles concentrados em tecnologia móvel. Como regra geral, os desenvolvedores de aplicativos não gostam de SOAP como linguagem de programação, e as ferramentas disponíveis para a criação de clientes de serviço web geralmente não oferecem suporte para a tecnologia móvel. Além das considerações práticas, há um problema de percepção: a organização que escolhe o estilo de serviço web pode ser considerada "antiquada", uma imagem que certamente diminuiria a adoção entre os desenvolvedores de aplicativos móveis.

É fácil perceber por que o estilo de REST pragmático se tornou tão popular. Como o URI é intuitivo e os desenvolvedores de aplicativos web e móveis geralmente estão familiarizados com as interfaces RESTful, é provável que a adoção e a produtividade dos desenvolvedores sejam altas. Além disso, a concentração no HTTP torna as APIs de REST pragmático ideais para o desenvolvimento dos aplicativos web e móveis de hoje. Atualmente, é provável que este seja o estilo preferencial para a maioria dos projetos.

No entanto, o estilo de REST pragmático não é perfeito para todos os contextos. É provável que os desenvolvimentos futuros desafiem sua predominância. Há vantagens e desvantagens distintas neste estilo: ele é limitado a quatro métodos, pode ser "loquaz" e a criação de URI não é padrão. Além disso, com a grande expansão da IoT (Internet of Things - Internet das Coisas) e do Big Data, e a consequente mudança na rede online, é provável que surjam desafios nesta abordagem centrada especificamente na web.

Hipermídia

O estilo de criação de API de hipermídia é uma abordagem com base em tarefas que visa oferecer uma alternativa mais sustentável para o REST pragmático. Como o REST pragmático, as APIs de hipermídia geralmente se concentram em padrões URI, HTTP e RESTful. Por um lado, porém, o estilo de hipermídia representa uma aplicação mais fiel da arquitetura RESTful, de acordo com Fielding, que descreve por que a web tem se mostrado tão expansível.

Dessa forma, a abordagem de hipermídia é ainda mais centrada na web: os hiperlinks e os formulários da web são espelhados na maneira em que uma API de hipermídia fornece links para navegar pelo fluxo de trabalho e pelas entradas de modelos para solicitar informações. Assim como a arquitetura

RESTful da web provou ser altamente expansível e passível de evolução, uma API de hipermídia criada de maneira adequada pode continuar fornecendo suporte a novos aplicativos por anos.

Embora essa abordagem de arquitetura seja claramente uma opção atraente para as empresas que pretendem criar APIs expansíveis e que forneçam um suporte confiável a aplicativos web e móveis em longo prazo, ela ainda se trata de um estilo de criação emergente com uma evidente falta de ferramentas associadas. Esse fato pode prejudicar as taxas de adoção dos desenvolvedores e dificultar o trabalho daqueles que adotam a API, impedindo que eles criem aplicativos cliente avançados com rapidez.

Orientado a eventos

Embora os estilos focados em HTTP, como REST pragmático e hipermídia, possam ser ideais para os aplicativos web e móveis que conhecemos atualmente, a chegada do HTML5 e da IoT está mudando esse cenário, possibilitando a criação de aplicativos mais dinâmicos, mas também exigindo interfaces mais leves. Neste contexto, o estilo orientado a eventos surgiu como uma alternativa indiferente quanto ao transporte, ideal para permitir que os aplicativos usem WebSockets e alternativas emergentes ao HTTP.

Este estilo, que se concentra em eventos iniciados pelo servidor ou pelo cliente, fornece uma opção de baixo custo, capaz melhorar o desempenho em cenários onde um grande número de pequenas mensagens passam entre o back-end e o aplicativo. Portanto, é ideal para IoT e uma série de casos

de uso de tecnologia móvel — principalmente sistemas de mensagens instantâneas, bate-papo com vídeo, jogos para vários jogadores e muitas outras opções. É provável que esse estilo também chame a atenção dos desenvolvedores mais vanguardistas.

É claro que nem todos os desenvolvedores desejam ser vanguardistas e que há muitos casos de uso em que uma abordagem RESTful convencional será mais apropriada. O HTTP ainda é o protocolo de transporte que domina a web, mas ele não lida muito bem com os eventos enviados pelo cliente. Além disso, o modelo de solicitação e resposta com base no qual este estilo foi criado dificulta o trabalho dos desenvolvedores de criar aplicativos cliente.

Figura 5: estilos de arquitetura para criação de APIs

 Serviço web	 REST pragmático	 Hipermídia	 Orientado a eventos
Disponibilidade de várias ferramentas relacionadas à SOA Não é adequado para a tecnologia móvel	Ideal para aplicativos web e móveis Conhecido pela maioria dos desenvolvedores de aplicativos Talvez não seja adaptável ao longo do tempo	Altamente centrado na web Expansível e passível de evolução Não é conhecido por muitos desenvolvedores	Apropriado para IoT e dispositivos Leve e dinâmico Não é adequado para cenários padrão

O estilo escolhido dependerá de suas restrições técnicas, dos objetivos de negócios e das preferências dos desenvolvedores. Tenha cuidado para não cair na armadilha de adotar um estilo "moderno", mas inadequado para seu contexto específico. Ao mesmo tempo, tente escolher um estilo que seja expansível e adaptável em longo prazo, à medida que seus recursos mudarem, seu público de usuários crescer e a natureza da rede online evoluir.

Independentemente do estilo que você escolher, existem determinados componentes de arquitetura que você desejará incluir em sua API. Na parte 6, descreveremos esses componentes e como eles serão organizados.

Parte 6: arquitetura de API

Os estilos de criação de arquitetura descritos anteriormente devem fornecer um modelo para a criação de um framework de arquitetura que ative a funcionalidade exclusiva da implementação da sua API. Determinados casos de uso exigirão a implementação de estilos específicos de criação. Também é importante observar, no entanto, que há uma série de componentes que devem ser incluídos em qualquer arquitetura de API, independentemente do caso de uso.

Esses componentes comuns de arquitetura não devem ser incorporados à implementação de determinada API. Em vez disso, eles devem ser implantados em uma infraestrutura principal da API que atuará como mediadora entre as APIs da organização e os aplicativos cliente que utilizam essas APIs. A concentração desses componentes em um único local torna mais rápido e fácil o processo de criar APIs adicionais, atualizar uma variedade de APIs de uma vez e garantir o bom funcionamento de APIs, sistemas de back-end e aplicativos cliente.

Para aumentar ao máximo a eficácia, esses componentes devem ser projetados em camadas, para que todo o tráfego de dados passe por cada uma das camadas indicadas à direita, na ordem especificada.



Figura 6: camadas de arquitetura



A camada de segurança

Além de abrir um leque de oportunidades de negócios, as APIs têm o potencial de expor a empresa a uma série de novas ameaças de segurança, por meio da divulgação de sistemas back-end e dados confidenciais para o mundo exterior. As APIs são vulneráveis a muitas das ameaças de segurança que assolam a web, além de enfrentar uma gama de novas ameaças específicas para as APIs. Portanto, é essencial implantar uma segurança de alto nível e específica para APIs na borda da sua arquitetura de API.

Essa necessidade de segurança de alto nível pode entrar em conflito com um objetivo básico da criação de APIs — uma API criada de maneira adequada facilita o trabalho dos desenvolvedores de criar aplicativos que forneçam um acesso direto aos recursos corporativos. É provável que a segurança de alto nível cause um impacto nessa facilidade de acesso. A implantação da segurança em uma arquitetura centralizada de API (em vez de aplicá-la na implementação da API) ajudará a mitigar esse impacto, além de permitir o uso de tecnologias flexíveis de gerenciamento de acesso, como OAuth e OpenID Connect.

A camada de cache

A eficiência da interface será essencial para proporcionar as experiências de desenvolvedor e de usuário final sem atritos que são necessárias para atingir os objetivos de adoção e de retenção do seu programa de API. Uma maneira de maximizar a eficiência da API é aplicar uma camada de cache perto da borda da arquitetura de API. Essa camada permite que as respostas armazenadas em cache sejam entregues para solicitações comuns, reduzindo a pressão colocada sobre as implementações e os recursos de back-end reais da API.

A camada de representação

É evidente que a apresentação da sua API deve ser a mais adequada possível para os desenvolvedores. Ao concentrar esse elemento longe da implementação, é possível se focar na criação de um acesso receptivo para suas APIs em um lugar central, sem prejudicar as APIs nem os recursos de back-end. Desse modo, é muito mais fácil apresentar sistemas de back-end complexos como interfaces web ou móveis que poderão ser compreendidas e utilizadas rapidamente pelos desenvolvedores para criar aplicativos avançados e fáceis de usar.

A camada de orquestração

Embora alguns aplicativos possam gerar valor acessando um único recurso por meio de uma única API, as possibilidades crescem exponencialmente quando você reúne dados de várias APIs (inclusive aquelas de outras empresas) e recursos de back-end. A implantação de uma camada de orquestração ao lado das interfaces possibilita essas combinações, além de simplificar o processo de compor novas implementações de vários recursos de back-end.

A maneira mais eficaz de criar uma arquitetura centralizada de API é implantar uma solução de gerenciamento de APIs. Na parte sete, descreveremos os principais componentes de gerenciamento de APIs.

Parte 7: gerenciamento de APIs

A criação de uma infraestrutura que centraliza os componentes comuns de arquitetura de APIs seguras e centradas nos desenvolvedores pode simplificar significativamente o processo de implementação de APIs que agregam valor real ao seu negócio. No entanto, a criação desse tipo de infraestrutura internamente pode ser um grande desafio. Felizmente, muitos fornecedores de software corporativo agora oferecem soluções de "gerenciamento de APIs" que eliminam a necessidade de desenvolver na empresa essa infraestrutura essencial.

Além disso, como o nome sugere, as soluções de gerenciamento de APIs também incluem uma funcionalidade para gerenciar e otimizar o desempenho das APIs em longo prazo. As soluções mais avançadas também têm recursos para a criação de uma interface com base na web, por meio da qual os desenvolvedores podem descobrir, conhecer e acessar as APIs — uma parte absolutamente essencial da apresentação de uma API centrada nos desenvolvedores, que não pode ser incorporada à implementação.

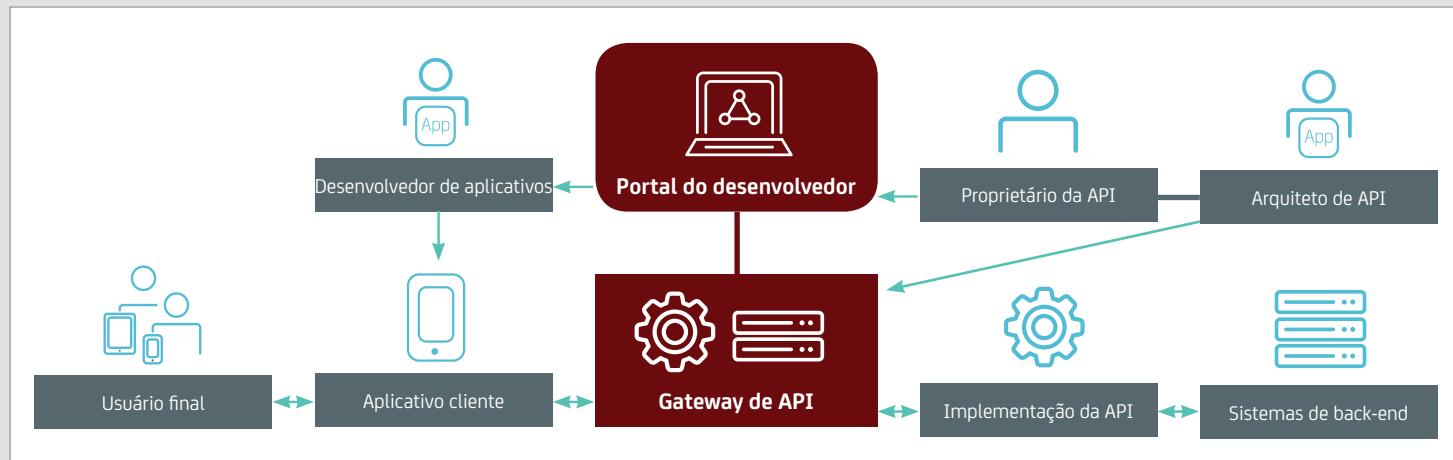


Componentes de gerenciamento de APIs

Uma solução de gerenciamento de APIs em nível corporativo terá dois componentes essenciais:

- Gateway de API – Oferece a segurança, o cache e a funcionalidade de orquestração que são necessários para implantar uma arquitetura de API principal.
- Portal do desenvolvedor – Fornece uma interface personalizável, por meio da qual os desenvolvedores acessam as APIs, bem como a documentação, os fóruns da comunidade e outros conteúdos úteis.

Figura 7: componentes de gerenciamento de APIs



É importante observar que o gerenciamento de APIs não é meramente um requisito técnico. A influência dele no sucesso comercial de qualquer programa de API corporativa é inevitável. O gerenciamento da composição, do desempenho e da segurança das APIs corporativas é essencial para garantir que a organização obterá um bom retorno sobre seu investimento em um programa de API. Da mesma forma, é essencial envolver e gerenciar ativamente os desenvolvedores para assegurar que eles criem aplicativos que geram valor comercial.

Para a maioria das empresas, uma infraestrutura de gerenciamento de APIs se tornará essencial para criar, implantar e manter as APIs que os desenvolvedores usarão para criar aplicativos novos realmente avançados.

[Descubra os princípios básicos do gerenciamento de APIs com o eBook 5 pilares do gerenciamento de APIs](#)

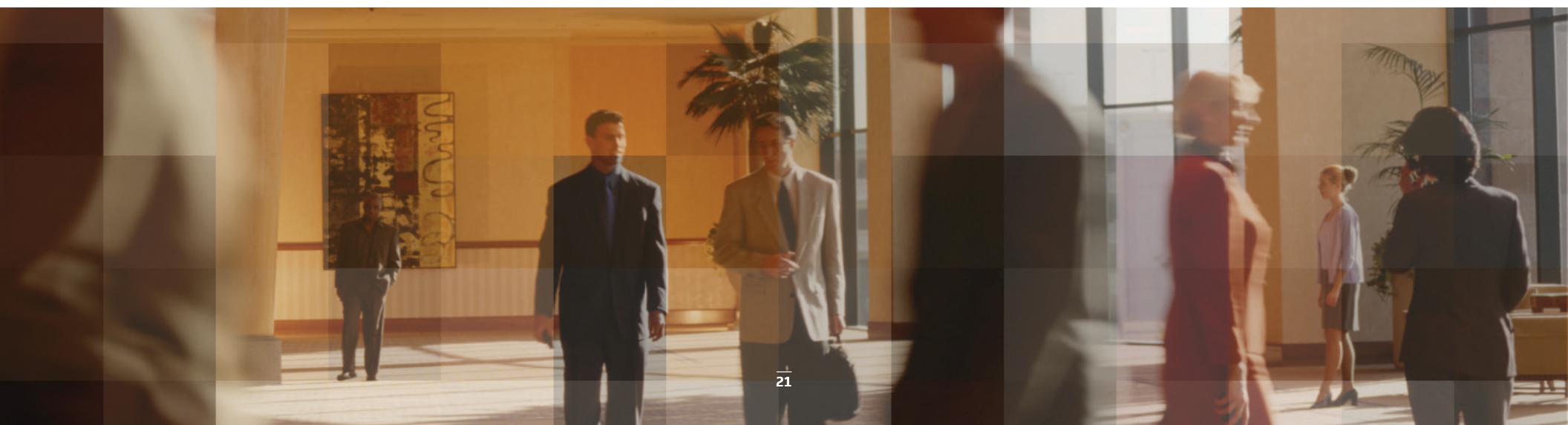
Conclusão

Do ponto de vista da arquitetura, as APIs representam uma extensão da SOA. Assim como a SOA criou interfaces para abrir sistemas herdados para reutilização em novos serviços que transpõem as fronteiras organizacionais, as APIs são usadas para abrir o back-end corporativo para os desenvolvedores que criam aplicativos para dispositivos móveis e a web pública. Essa é uma extensão significativa, por isso, os requisitos de criação para uma API da web provavelmente serão muito diferentes daqueles de um serviço web de SOA.

Embora os programas de SOA geralmente sejam direcionados pela necessidade de reduzir os custos de TI, os programas de API se concentram em gerar novos fluxos de receita. Uma API da web conecta uma variedade de ativos de negócios existentes a fim de gerar valor de formas inesperadas. Uma boa criação de APIs está sempre focada nos resultados comerciais. Portanto, as práticas de arquitetura e criação de APIs devem estar alinhadas com a estratégia comercial da organização, do início ao fim.

Os arquitetos e proprietários de APIs devem se comunicar para garantir que estão de acordo sobre os principais objetivos, o que será feito para atingi-los e a maneira de avaliar o sucesso. Para garantir uma comunicação eficaz, um evangelista de API capaz de preencher a lacuna entre as funções técnicas e de negócios deve analisar as necessidades dos líderes de negócios, proprietários de API, desenvolvedores de aplicativos e arquitetos corporativos para estabelecer um conjunto apropriado de metas, tarefas e métricas.

Na prática, a criação de uma API para o sucesso dos negócios geralmente significa projetar uma interface que os desenvolvedores realmente desejam usar. Portanto, antes de criar algo, é essencial pesquisar sistematicamente seu público de desenvolvedores para compreender quem são eles e o que eles querem de uma API. Também pode ser útil testar as premissas sobre as preferências dos desenvolvedores, oferecendo protótipos leves de APIs.



Quando você estiver pronto para criar sua verdadeira implementação da API, deverá escolher o estilo de criação que melhor se adapte ao seu projeto. As APIs de serviço web serão adequadas para programas internos destinados a desenvolvedores com experiência em SOA. As APIs de REST pragmático são mais adequadas para projetos de API aberta, focados em dispositivos móveis e na web. Os estilos de hipermídia e orientado a eventos estão emergindo como abordagens que talvez sejam mais sustentáveis no futuro, com a tecnologia móvel e a IoT.

Seja qual for o estilo, há determinados elementos de arquitetura que todas as APIs devem incluir — ou seja, segurança, cache, representação e orquestração. Para aumentar ao máximo a eficácia e a capacidade de gerenciamento, esses elementos não devem ser incorporados às implementações individuais de APIs. Em vez disso, todas as APIs devem utilizar uma arquitetura de API central e em camadas, que atuará como mediadora entre a empresa e as APIs.

A forma mais eficiente e eficaz de implantar uma arquitetura de API central — e garantir o sucesso do programa de API em longo prazo — é adotar uma solução de gerenciamento de APIs. Há uma variedade de soluções no mercado, mas a maioria inclui dois componentes comuns:

- Um gateway de API que fornece a funcionalidade de segurança e outras infraestruturas importantes.
- Um portal do desenvolvedor que simplifica o processo de envolver e capacitar os desenvolvedores.

Há muito em jogo nos projetos de API corporativa da atualidade — grandes oportunidades de negócios, riscos de segurança significativos, entre outras questões. É essencial que você se prepare antes de começar a criar uma API: alinhe os objetivos de criação com os objetivos de negócios; estabeleça quais são as preferências do seu público de desenvolvedores; escolha um estilo de implementação apropriado; e implante uma infraestrutura de gerenciamento de APIs. Assim, você estará pronto para criar uma API realmente valiosa.

Figura 8: pré-requisitos de uma boa criação



Somente o CA API Management permite que as organizações integrem sistemas, simplifiquem o desenvolvimento de aplicativos e rentabilizem os dados com o nível de segurança e proteção de API de que as empresas precisam hoje. Saiba mais sobre o CA API Management em ca.com/br/api

Sobre o CA API Management

Com mais de 300 clientes de gerenciamento de APIs em setores tão diversos como comunicação, serviços financeiros, governo e varejo, a CA Technologies oferece a tecnologia líder do setor e o know-how que ajudam as organizações a gerarem valor por meio das APIs. A CA fornece uma solução completa de gerenciamento de APIs, incluindo um gateway de API totalmente funcional, com recursos de segurança de nível militar, além de um portal do desenvolvedor em versões de SaaS e no local. Saiba mais sobre o CA API Management em ca.com/br/api.

API Academy

Serviços de estratégia, arquitetura e criação de APIs

A equipe da API Academy consiste em especialistas do setor que foram reunidos pela CA Technologies para desenvolver recursos gratuitos para a comunidade e fornecer serviços especializados de consultoria para organizações que desejam elevar seus programas de API a um novo patamar. Para saber como a API Academy pode ajudar sua organização com a estratégia, a arquitetura e a criação de APIs, visite apiacademy.com.

A **CA Technologies** (NASDAQ: CA) cria software que acelera a transformação das empresas e permite que elas aproveitem as oportunidades da economia dos aplicativos. O software está no cerne de todas as empresas, em todos os setores. Do planejamento ao desenvolvimento e do gerenciamento à segurança, a CA está trabalhando com empresas de todo o mundo para mudar a maneira como vivemos, fazemos negócios e nos comunicamos – usando dispositivos móveis, as nuvens privada e pública e os ambientes distribuídos e de mainframe. Obtenha mais informações em ca.com/br.

Copyright © 2015 CA. Todos os direitos reservados. Todas as marcas comerciais, nomes de marcas, marcas de serviço e logotipos aqui mencionados pertencem às suas respectivas empresas. Este documento destina-se apenas a fins informativos. A CA não assume responsabilidade pela precisão ou integridade das informações. Na medida do permitido pela lei aplicável, a CA fornece este documento "no estado em que se encontra", sem garantias de nenhum tipo, incluindo, sem limitações, garantias implícitas de comercialização, adequação a uma finalidade específica ou não violação. Em nenhuma circunstância a CA será responsável por perdas ou danos, diretos ou indiretos, decorrentes do uso deste documento, incluindo, sem limitações, perda de lucros, interrupção de negócios, reputação da empresa ou perda de dados, mesmo que a CA tenha sido expressamente informada sobre a possibilidade de tais danos com antecedência.

CS200-131275

