



SOA: Desafios e Soluções

Visão Geral Os benefícios da SOA (Arquitetura Orientada a Serviços) foram bem documentados, mas os desafios associados à sua implementação por toda uma empresa, ainda não. Embora a natureza distribuída da SOA favoreça a reutilização e crie um alto nível de agilidade para a empresa, também pode incorrer em desafios concretos para fornecer aplicações baseadas na SOA.

XML é a tecnologia que permite as aplicações baseadas na SOA. Sua natureza prolixa, sua inerente falta de segurança e a crescente necessidade de conexões entre aplicações e serviços resultam em vários desafios já bastante estudados e com soluções comprovadas.

Desafio O desafio de fornecer aplicações baseadas na SOA está em identificar os possíveis problemas que podem surgir e resolvê-los o mais cedo possível no ciclo de implementação. Controles de fornecimento de aplicação são adequados para enfrentar problemas conhecidos ou inesperados associados ao fornecimento de aplicações baseadas em SOA. Incorporar um application delivery controller (ADC) nos planos de implementação de SOA pode prevenir atrasos desnecessários e mesmo a necessidade de ter que rearquitar partes da sua infra-estrutura SOA – poupando horas de trabalho, despesas e dores de cabeça.

Solução SOA é tanto um padrão de design arquitetônico como uma metodologia de implementação. Essa é uma boa maneira de dizer que ninguém consegue definir como uma implementação SOA em particular deveria se parecer. Embora haja tentativas de vários fornecedores em alguns mercados verticais de SOA para criar as melhores práticas para ambientes SOA, elas falharam grandemente devido à natureza um tanto dinâmica e específica à organização das implementações SOA. Como os serviços que incluem SOA são focados na empresa e encapsulados por entidades corporativas específicas, não há uma definição consensual do que precisa existir e quais componentes devem ser empregados para se poder utilizar o título de SOA.

Entretanto, existe um conjunto de princípios básicos da SOA que pode ser considerado referência para as discussões dos atributos e desafios comuns da SOA. A natureza distribuída dos serviços, por exemplo, é um atributo fundamental de todas as implementações SOA e, portanto, os desafios associados a esse modelo de implementação podem ser esperados em todos os ambientes SOA, sejam quais forem o design e a implementação reais.

1. SOA engloba serviços corporativos distribuídos e alcança valor comercial por meio da reutilização desses serviços.
2. SOA baseia-se em padrões abertos aceitos no setor.
3. SOA reduz o tempo de lançamento no mercado por meio da união das interfaces de serviço e suas implementações subjacentes, ganhando agilidade.
4. SOA é heterogênea e tem n-camadas, englobando aplicações que podem variar segundo a atividade dos processos corporativos que originaram os serviços combinados.

Esses atributos comuns resultam em um conjunto de desafios comuns que todas as organizações encontram e que devem ser enfrentados em algum momento do processo de implementação. Os princípios que guiam a SOA sugerem considerar a arquitetura do ambiente e da infra-estrutura de implementação nos quais os serviços

serão empregados antes que eles entrem no estágio de produção, em paralelo com a definição do serviço e com os esforços de implementação.

Felizmente, a maioria dos desafios associados à implementação da SOA é comum a todas as implementações de aplicações web. A diferença entre as aplicações tradicionais e as baseadas na SOA é que os desafios destas são enfrentados pelos implementadores mais cedo no ciclo de vida da aplicação devido aos desafios associados à sua tecnologia habilitadora central, a XML.

Fundamentos da SOA

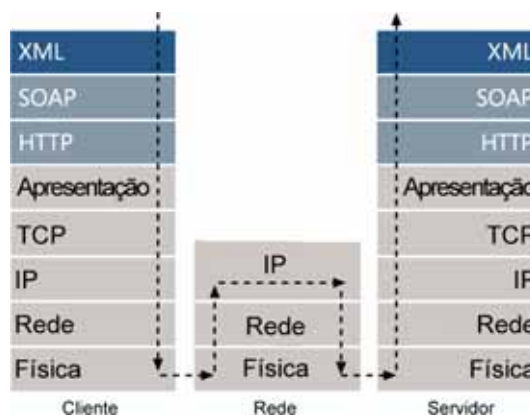
Na maioria dos casos, uma SOA será implementada usando vários padrões, sendo o mais comum o HTTP, o WSDL (Linguagem de Definição de Serviços Web), o SOAP (Protocolo Simples de Acesso a Objeto) e a XML (Linguagem de Marcação Extensiva). Esses últimos três padrões funcionam em conjunto para distribuir mensagens entre serviços de um modo muito parecido com um posto de correio.

WSDL	→	Entrada em um catálogo de endereço
HTTP	→	Carteiro (transporte)
SOAP	→	Envelope (encapsulamento)
XML	→	Carta (mensagem)

Se quiser enviar uma carta, assume-se que você já sabe para onde. Na SOA, essa informação vem do WSDL, que lista as opções de transporte (HTTP, FTP, SMTP) disponíveis, os possíveis endereços (por meio de funções remotas) e as exigências de formato para o dado enviado a cada endereço. Da mesma forma que um envelope tem um endereço, o SOAP leva em si a informação sobre onde a mensagem deverá ser entregue e quem poderá abri-la. No caso do SOAP e da SOA, o endereço não é um *quem*, mas um *o quê*, já que o destinatário é, na verdade, o nome de uma função remota que deverá processar a mensagem interna. Assim como um envelope “real”, o SOAP precisa seguir certo formato e, se não colocar a informação correta no lugar certo, haverá problemas de entrega ou processamento.

Com essa informação, é possível criar uma mensagem baseada nas exigências de formato (baseada na XML), colocar em um envelope endereçado ao destinatário (SOAP) e então pôr na caixa de correio (HTTP) e esperar a resposta.

Se você for o receptor dessa mensagem, ela precisa ser entregue a você, você precisa ler (avaliar) o envelope e determinar o destinatário correto (a função remota). A mensagem, então, precisa ser passada para o destinatário apropriado (função remota) para ser processada. Assim que o destinatário correto tiver a carta, ele pode, então, lê-la (avaliar) e processá-la segundo a função designada.





Observe que esse processo requer bastante leitura (avaliação) e formatação dos dados, do dado real que se quer transferir até o envelope para a camada de transporte. Ele também tem o efeito de adicionar duas “camadas” à pilha a ser processada. Esse é um dos desafios associados com a SOA – a sobrecarga computacional associada com a avaliação e processamento da XML. As análises atuais do impacto da avaliação e processamento da XML na utilização do servidor estimam que aproximadamente 30% dos recursos de um servidor são consumidos simplesmente para avaliar e processar XML. Como ponto de referência, também foi estimado que o processamento consome 30% dos recursos de um servidor. Diferente do SSL, a XML atualmente não tem uma solução eficaz baseada em servidor para aliviar esse fardo como um processamento assistido por hardware, portanto a questão de melhorar o desempenho dos serviços e aplicações baseados em XML deve estar necessariamente fora do servidor ou da rede.

Obviamente, uma única mensagem SOAP não faz uma SOA. Uma SOA pode ser pensada como uma organização distribuída que faz uso do serviço postal (rede) para entregar mensagens que designam tarefas (funções remotas) às unidades corporativas individuais (serviços).

Também não há nada que impeça que os serviços sejam construídos segundo outra tecnologia. A REST (Representational State Transfer) é geralmente usada como exemplo de uma tecnologia alternativa na qual a SOA pode ser construída. Os serviços REST não usam SOAP, mas sim HTTP URI para especificar o destinatário da mensagem e colocar a mensagem direto no corpo HTTP. Se o SOAP é análogo à carta, então as mensagens REST são mais como um cartão postal onde a mensagem nem sequer é encapsulada.

Uma SOA pode misturar e combinar ambas as tecnologias, assim como outras, embora as poucas melhores práticas documentadas e os consultores de SOA sejam rápidos em empurrar – e até mesmo impor – às organizações uma ou outra padronização. Muitos dos benefícios da SOA, incluindo a interoperabilidade e a independência de plataforma, apóiam-se na fundação de padrões abertos, e esse movimento não é necessariamente ruim.

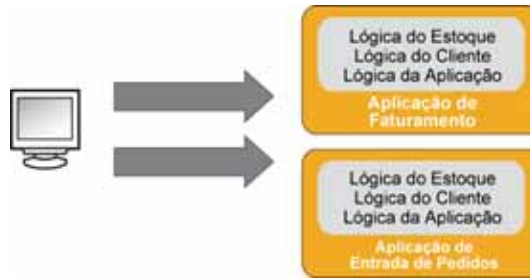
Gerenciamento de Conexão

Arquitetar os serviços distribuídos sobre os quais uma SOA é construída envolve a identificação das entidades corporativas comuns e suas funções correspondentes que podem ser encapsuladas como um serviço. O estoque de uma empresa, por exemplo, é uma entidade corporativa comum que provavelmente é compartilhada entre todas as unidades da empresa. Pesquisar e atualizar um estoque são duas funções comuns realizadas por aplicações no estoque. Da mesma forma, é provável que exista uma entidade de cliente comum a todas as aplicações na organização, com um conjunto comum de funções que pode ser realizado naquele cliente, como pesquisar, atualizar ou criar.

No passado, essas funções e entidades comuns freqüentemente eram duplicadas entre todas as aplicações, sendo preciso aplicar a elas a lógica de cada aplicação. Isso significava que necessariamente qualquer mudança nessas entidades implicava modificar qualquer aplicação que as usasse. Esse é um modelo ineficiente, que deixa em aberto a possibilidade bastante plausível de que haverá diferenças em como as aplicações manipulam essas entidades. Essas diferenças são replicadas na armazenagem de dados, gerando problemas entre aplicações díspares que precisam acessar esses dados para realizar tarefas específicas. A duplicação de funções entre múltiplas aplicações também tem um efeito danoso na capacidade de uma



organização em mudar rapidamente, devido ao tempo e ao esforço envolvido em implementar uma mudança em qualquer entidade compartilhada.



Design Tradicional de Aplicação

A SOA resolve essas questões removendo as entidades comuns e suas funções associadas e estabelecendo-as como uma entidade atômica independente (um serviço), que é compartilhada entre todas as aplicações.

As mudanças na lógica e na estrutura de dados associada àquela entidade agora ficam restritas a um único conjunto de código, e a consistência é assegurada para toda a empresa.



Design SOA de Aplicação

Embora uma SOA certamente melhore a consistência da lógica da aplicação e reduza o tempo necessário para se introduzir uma mudança nas entidades corporativas ou na suas estruturas de dados subjacentes, pode-se observar que ela aumentou a complexidade em termos de números de conexões exigidas.

O aumento nas conexões requeridas pode ter um impacto negativo no desempenho se os serviços estiverem fisicamente distribuídos pela empresa. A sobrecarga associada ao gerenciamento das conexões TCP é aplicada diretamente ao custo computacional da aplicação, geralmente em detrimento do desempenho geral da aplicação. Observe também que cada intermediário (isto é, dispositivo ou aplicação que toque uma mensagem XML) incorre em penalidades de desempenho tanto pela sessão TCP quando pela análise XML. Conforme cresce o número de serviços associados a uma única aplicação, também aumenta o tempo de resposta total da aplicação.

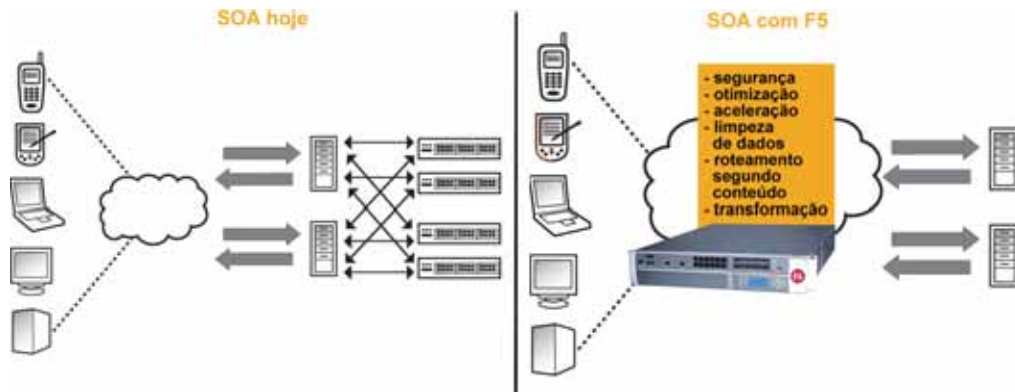
Essencialmente, há duas maneiras pelas quais os ADCs podem abordar esse desafio. A primeira se baseia nas consagradas e testadas tecnologias de gerenciamento de conexão. Ao permitir que um ADC gerencie a miríade de conexões entre os serviços, alivia-se o fardo de gerenciar as sessões TCP nos servidores, e a capacidade total cresce.



Tecnologia de gerenciamento de conexões reduz o ônus do TCP nos servidores

A segunda maneira dos ADCs enfrentarem os desafios dos serviços distribuídos é fornecendo uma plataforma de rede na qual os serviços de rede compartilhados possam ser hospedados.

Muitas dessas funções – segurança, limpeza e transformação de dados, autenticação e cache – são específicas da aplicação e, portanto, as soluções pontuais focadas na rede não podem hospedar esses serviços compartilhados. Estes só podem ser suportados por um ADC ou uma plataforma de aplicação como as oferecidas por BEA, IBM, Oracle ou Microsoft.



Mover os serviços compartilhados focados em rede adequados a um ADC reduz a complexidade da SOA, pois consolida os serviços em uma única plataforma hospedada em rede sem sacrificar o desempenho. Hospedar os serviços de rede compartilhados em um ADC também traz o benefício de aliviar dos serviços da sua SOA as análises de XML desnecessárias. Ao validar as credenciais das mensagens que chegam antes de alcançar o serviço, o serviço só precisa analisar aquelas mensagens que estão autorizadas a invocar suas operações. Esse princípio se aplica igualmente bem tanto com XML quanto com as medidas de defesa contra ameaça de aplicações web. Em vez de obstruir o desempenho duplicando as verificações contra ameaças comuns, como as injeções de SQL, os tamanhos de mensagem excessivamente grandes, o manejo de cookies e os ataques específicos de XML, faz sentido fornecer um único serviço compartilhado por meio do qual a mensagem é inicialmente validada e verificada se não contém conteúdo malicioso.

Verificar a mensagem no ponto de ingresso alivia a necessidade de cada serviço realizar a mesma verificação e, portanto, reduz a carga de processamento nos seus respectivos servidores e aumenta a capacidade geral.



Escalonamento

O alto custo computacional de análise e processamento de XML reduz a capacidade geral dos já carregados servidores de aplicação. O problema com a XML surge da sua formatação de texto, o qual precisa ser analisado e convertido em objetos que o servidor de aplicação possa entender e manipular. Esse processo é chamado de marshalling (literalmente, organização) e o desempenho desse processo depende bastante da complexidade dos dados. Cada elemento dos dados precisa ser criado e muitas funções são chamadas para designar valores a seus atributos. Na maioria dos ambientes, esse processo é realizado por serialização, na forma de entradas e saídas dentro do sistema. As entradas e saídas em geral usam intensivamente os recursos, querem muita memória e ciclos da CPU no servidor independente da plataforma.

Esse processo ocorre duas vezes, uma quando a mensagem é recebida e transformada da XML para um formato legível para a máquina e, novamente, quando a resposta é transformada de volta em XML para ser encaminhada.

Por causa do dreno de recursos gerado por esse processo e por qualquer processamento adicional requerido, tal como as conexões e pesquisas nas bases de dados, os servidores de aplicação nos quais os serviços são implementados atingem rapidamente níveis elevados de utilização de CPU e o pool de memória disponível minguia. Isso reduz a capacidade dos servidores de aplicação de processar as requisições em tempo hábil e gera atrasos quando o sistema tenta processar criteriosamente cada requisição compartilhando seus recursos entre elas.

O efeito da alta carga nos servidores de aplicação é um problema bem estudado, com soluções conhecidas, como a assistência de ADCs no escalonamento horizontal desses servidores para distribuir as requisições.

É importante escalonar servidores de aplicações em um ambiente SOA logo no início do processo de implementação por causa da carga adicional posta nos servidores de aplicação pela análise e processamento de XML.

Não são somente os servidores de aplicação em um ambiente SOA que precisam escalonar os serviços.

Muitas organizações adotaram gateway appliances de XML tais como as da Reactivity e DataPower da IBM para aliviar o forte processamento de XML e fornecer outras funcionalidades específicas à XML, tais como segurança de serviços web, validação de esquemas, virtualização de serviços e transporte via XSLT (linguagem de folhas de estilo extensível para transformação). Esses dispositivos são capazes de realizar essas funções nas mensagens baseadas na XML com muito mais eficiência do que os servidores de aplicação, mas não têm muitos dos recursos de classe corporativa, como um algoritmo avançado de equilíbrio de carga, métodos comprovados contra falhas e gerenciamento de sessões.

Para escalonar esses dispositivos, ainda é necessário um ADC. Deve-se tomar o cuidado de considerar quando surge a questão do balanceamento de carga e as appliances XML. Embora esses dispositivos sejam capazes de balancear a carga, suas implementações são rudimentares e inflexíveis nas suas opções de implantação. Esses dispositivos não fornecem capacidades avançadas de verificação da saúde dos ADCs e, portanto, podem falhar em direcionar corretamente o tráfego, já que não têm consciência do estado da aplicação, como no caso de um ADC.

Os tradicionais algoritmos *round-robin* (de alternância) e menor-conexão usados nos clusters tanto de appliances quanto de servidores de aplicações XML falham em reconhecer a natureza consumidora de recursos da XML e, portanto, são incapazes de distribuir as requisições entre um pool ou um cluster de servidores de forma



realmente eficiente. A capacidade de um servidor de aplicação de manipular a carga de mensagens SOA depende não apenas do número de requisições sendo processadas, mas também dos recursos disponíveis no momento. Isso significa que são necessários algoritmos e capacidades avançadas para determinar o estado atual da aplicação e do servidor, o que é um dos domínios de especialização dos ADCs. Um servidor pode ter apenas uma conexão e ser o “próximo da fila” a receber uma requisição, mas, se estiver analisando e processando uma mensagem XML muito grande, poderá não ter memória nem ciclos de CPU para processar outra mensagem e ainda assim atingir os níveis de serviço acordados para ambas as mensagens.

Segurança

A XML é baseada em texto e, portanto, legível para uma pessoa. Ela também se baseia na Web e é hospedada na mesma infra-estrutura de aplicação, como as aplicações “tradicionais” baseadas na Web, tornando-a sujeita aos tradicionais ataques na Internet.

Muitas das mesmas vulnerabilidades que atormentam as aplicações web são aplicáveis às aplicações baseadas em XML. A injeção SQL é um método comum de tentar extrair informações não autorizadas de uma base de dados corporativa ou de criar uma destruição generalizada apagando dados importantes. A injeção SQL pode ser facilmente realizada por meio de uma mensagem XML via uma tradicional mensagem HTTP no formato GET e POST.

Como esse tipo de ataque é bem conhecido, existem soluções desenhadas para proteger as aplicações web. Essas mesmas soluções fornecem medidas de segurança para as aplicações baseadas em XML, assim como outras técnicas de segurança, como os mecanismos de verificação de assinatura que procuram descobrir vírus nas mensagens transmitidas pela Internet.

Embora sempre seja preferível que os desenvolvedores produzam os códigos de segurança, a verdade é que, se eles responderem a cada ameaça com um código, eles passarão a maior parte do tempo codificando a proteção contra novas ameaças, tentando essas soluções e implementando-as, somente para começar tudo de novo no dia seguinte. Além disso, tais técnicas de segurança em código requerem a duplicação do código por toda a aplicação, o que pode ter um impacto negativo no desempenho e aumentar a possibilidade de outros erros serem introduzidos no código. A duplicação do código é uma anátema nos princípios que guiam a SOA, que procura reduzir os custos e o tempo de lançamento das soluções por meio da identificação desses serviços compartilhados.

Empregando um firewall de aplicação para minimizar os riscos associados à SOA e aos ambientes de serviços distribuídos, os princípios de uma SOA se mantêm e protegem as aplicações desses tipos de ameaças bem conhecidas. Mover o máximo possível da segurança de aplicação para o ponto de ingresso traz o benefício adicional de melhorar o desempenho, já que as mensagens claramente mal intencionadas nunca atingem o servidor de aplicação, reduzindo, portanto, a carga na infra-estrutura de aplicação.

Largura de banda

Uma preocupação comum em relação ao XML é o aumento do tamanho das mensagens passando entre sistemas. De fato, o envelope SOAP usado pelos serviços web para carregar as mensagens XML adiciona por si só no mínimo mais 256 bytes a cada mensagem, e geralmente o aumento é muito maior.

A natureza prolixa da XML implica que mesmo as menores mensagens ficarão aproximadamente com o dobro do tamanho do que se fossem formatadas nos pares POST/GET em HTTP. Quando essas mensagens são transferidas entre servidores de um data center, geralmente o aumento no consumo de banda e no tempo de transferência é insignificante, devido à natureza grande dos backbones do data center. Mas, com o aumento do uso do JavaScript Assíncrono e XML (AJAX) como base das interfaces de usuários que interagem com sistemas back-end baseados em SOA, torna-se importante considerar o efeito dessas mensagens cada vez maiores no tempo de resposta ao cliente.

Além da possibilidade de clientes AJAX acessarem serviços SOA, há a possibilidade bastante real de que os parceiros externos e os clientes possam Acessar esses serviços por meio da Internet pública, sobre uma rede pública incontrolável e geralmente imprevisível. Em qualquer caso em que os clientes acessem os serviços via uma rede pública, há uma possibilidade do cliente não ser capaz de receber as respostas tão rapidamente quanto o servidor as envia. Isso força os servidores a segmentar os dados em pequenos blocos para que cliente possa consumir mais facilmente, mas aumenta o tempo total necessário para fornecer o conteúdo.

Isso amarra a aplicação e o servidor web, reduzindo o total de clientes que podem ser suportados em um dado momento. Os application delivery controllers resolvem essa questão empregando uma técnica chamada freqüentemente de “*content spooling*”. Ele permite a um ADC fazer a mediação entre o cliente e o servidor e permite ao servidor enviar o conteúdo o mais rápido possível ao dispositivo mediador. O ADC toma para si a responsabilidade de passar o conteúdo a contatogotas ao cliente, permitindo que a aplicação ou o servidor web lide com mais requisições.



Outro desafio que surge com o uso de XML no transporte de dados é que as mensagens são geralmente excessivamente grandes. Mensagens com 100 KB não são incomuns e, mesmo que raras, podem existir mensagens atingindo os gigabytes.

Com a solução certa é fácil enfrentar as questões associadas ao tamanho do HTML e de outros formatos de mensagens baseadas em texto. Existem várias tecnologias de aceleração capazes de empregar várias técnicas de compressão para reduzir o tamanho das mensagens, diminuindo assim o tempo de transferência entre o servidor e o cliente. Existem técnicas de compressão padrões no setor, de eliminação dos espaços em branco e outras tecnologias de redução de dados, somente para responder a esse desafio e reduzir o impacto das grandes mensagens na infra-estrutura de aplicação e na rede de fornecimento.

Os produtos específicos de aceleração de aplicação, assim como muitos dos recursos associados ao ADC, podem fornecer muitas dessas capacidades, bem como mecanismos avançados de assistência a questões encontradas nas aplicações baseadas em navegadores, como os limites de conexão artificialmente impostos que impedem que o AJAX e as tecnologias baseadas em XML atinjam o desempenho máximo. O cache dinâmico de conteúdo é aplicável ao XML da mesma forma que é no HTML e nos formatos baseados em texto e pode aumentar drasticamente o desempenho e a capacidade das implementações baseadas na SOA. Da mesma



forma, os controladores de otimização da WAN, aproveitando-se das vantagens das capacidades de redução de dados simétricas, podem acrescentar melhorias aos cenários de escritórios remotos.

Conclusão

Muitos dos desafios associados às implementações da SOA são os mesmos desafios que encontramos quando empregamos as aplicações tradicionais baseadas na web. As causas subjacentes desses desafios podem diferir em uma SOA, mas as soluções que as enfrentam continuam as mesmas. A chave para enfrentar os desafios relacionados à SOA é identificá-los o mais cedo possível e incluir a solução apropriada na arquitetura para reduzir a possibilidade de ter de rearquitar um ambiente pós-implementado.

Desafio	Impacto	Solução ADC	Benefícios
Aumento exponencial das conexões	<ul style="list-style-type: none"> • Sobrecarga o gerenciamento das conexões TCP aumenta o fardo sobre os servidores • Saltos adicionais criam maior latência 	<ul style="list-style-type: none"> • Gerenciamento de conexão • Hospedagem dos serviços compartilhados 	<ul style="list-style-type: none"> • Reduz o número de servidores necessários, diminuindo custos • Reduz a complexidade e o tempo de gerenciamento • Mantém a disponibilidade e aumenta o desempenho • Reduz a necessidade de duplicação de funcionalidades nos serviços implementados
Tamanho das mensagens XML	<ul style="list-style-type: none"> • Maior consumo de banda • Maior consumo de recursos para analisar/processar as mensagens 	<ul style="list-style-type: none"> • Mediação do conteúdo entre cliente-servidor (content spooling) • Compressão • Cache • Hospedagem dos serviços compartilhados • Qualidade de serviço 	<ul style="list-style-type: none"> • Aumenta a capacidade do servidor, reduzido custos e aumentando a eficiência operacional • Minimiza os riscos de ataque • Prioriza as mensagens, assegurando o cumprimento do nível de serviço acordado
Escalonamento	<ul style="list-style-type: none"> • A análise XML requer computação intensiva e infra-estrutura horizontalmente escalonável • As appliances XML que aliviam as tarefas específicas de XML não suportam a persistência, antifalha ou o balanceamento avançado de carga 	<ul style="list-style-type: none"> • Os ADCs suportam os cenários de persistência, balanceamento avançado de carga e antifalhas 	<ul style="list-style-type: none"> • Mantém a disponibilidade dos serviços • Aumenta o desempenho • Protege os investimentos em tecnologia, assegurando o escalonamento horizontal • Suporta as capacidades essenciais que as outras soluções não têm
Segurança	<ul style="list-style-type: none"> • Exploração das vulnerabilidades da infra-estrutura de aplicação • Roubo de dados confidenciais 	<ul style="list-style-type: none"> • Capacidades de firewall de aplicação • Limpeza de conteúdo • Validação de dados 	<ul style="list-style-type: none"> • Previne que ataques atinjam a infra-estrutura de aplicação • Melhora o desempenho da infra-estrutura de aplicação reduzindo a duplicação de código

**Sobre a F5**

A F5 Networks é a líder global em Application Delivery Networks. A F5 fornece soluções que tornam os aplicativos seguros, rápidos e disponíveis para todos, ajudando as companhias a obter o maior retorno pelo seu investimento. Ao implementar inteligência e gerenciabilidade na rede para transferir a carga de aplicativos, a F5 os otimiza, permitindo que eles trabalhem mais rápido e consumam menos recursos. A arquitetura expansível da F5 integra de forma inteligente a otimização de aplicativos, protege os aplicativos e a rede e oferece confiabilidade aos aplicativos - tudo em uma plataforma universal. Mais de 10.000 companhias e provedores de serviços em todo o mundo confiam na F5 para manter seus aplicativos funcionando. A companhia tem sede em Seattle, Washington, com escritórios no mundo todo. Para mais informações, visite www.f5.com (em inglês).