Introdução às tecnologias Web Services: SOA, SOAP, WSDL e UDDI - Parte1

Neste artigo veremos uma introdução às tecnologias Web Services: SOA, SOAP, WSDL e UDDI.

Esse artigo faz parte da revista WebMobile edição 1. Clique aqui para ler todos os artigos desta edição.

Antes de nos aprofundarmos nos conceitos e tecnologia de web services, vejamos um pouco sua evolução. No ano de 2000, a W3C (World Wide Web Consortium) aceitou a submissão do Simple Object Access Protocol (SOAP). Este formato de mensagem baseado em XML estabeleceu uma estrutura de transmissão para comunicação entre aplicações (ou entre serviços) via HTTP. Sendo uma tecnologia não amarrada a fornecedor, o SOAP disponibilizou uma alternativa atrativa em relação aos protocolos proprietários tradicionais, tais como CORBA e DCOM.

No decorrer do ano seguinte, o W3C publicou a especificação WSDL. Uma nova implementação do XML, este padrão forneceu uma linguagem para descrever a interface dos web services. Posteriormente suplementada pela especificação UDDI (Universal Description, Discovery and Integration), que proporcionou um mecanismo padrão para a descoberta dinâmica (dynamic discovering) de descrições de serviço, a primeira geração da plataforma de Web services foi estabelecida.
A **Figura 1** ilustra em alto nível o relacionamento entre estes padrões.

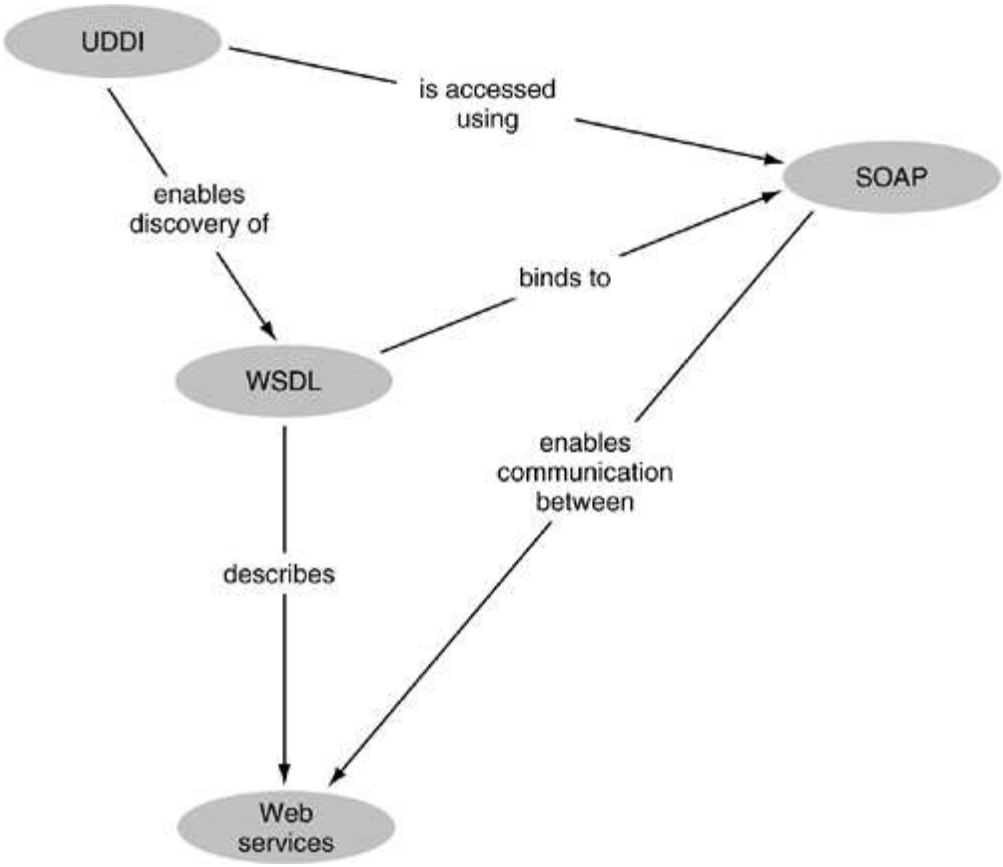

Figura 1: O relacionamento entre especificações de primeira geração

- Is accessed using: é acessado utilizando;
- Enables discovery of: permite a descoberta de;
- Describes: descreve;
- Enables communication between: permite a comunicação entre;
- Binds to: ligação para.

Desde então, os web services foram adotados por vendedores e fabricantes num ritmo considerável. Suporte amplo da indústria seguiu-se à popularidade e importância desta plataforma e de princípios de projeto orientados a serviço. Isto levou à criação de uma segunda geração de especificação de Web services.

Web services e a arquitetura orientada a serviços (SOA)

Entendendo serviços

O conceito de serviços em uma aplicação existe faz algum tempo. Serviços, assim como componentes, são considerados blocos de construção independentes, os quais coletivamente representam um ambiente de aplicação. No entanto, diferente de componentes tradicionais, serviços têm algumas características únicas que lhes permitem participar como parte de uma arquitetura orientada a serviços.

Uma destas características é a completa autonomia em relação a outros serviços. Isto significa que cada serviço é responsável por seu próprio domínio, o que tipicamente significa limitar seu alcance para uma função de negócio específica (ou um grupo de funções relacionadas).

Este enfoque de projeto resulta na criação de unidades isoladas de funcionalidades de negócio ligadas fracamente entre si. Isto é possível por causa da definição de uma estrutura padrão de comunicação. Devido à independência que esses serviços desfrutam dentro desta estrutura, a lógica de programação que encapsulam não tem necessidade de obedecer a nenhuma outra plataforma ou conjunto de tecnologias.

XML Web services

O tipo de serviço mais largamente aceito e bem sucedido é o XML Web service, que será daqui em diante chamado apenas de web service, ou simplesmente service. Este tipo de serviço possui dois requisitos fundamentais:

- comunica-se via protocolos internet (normalmente HTTP);
- envia e recebe dados formatados como documentos XML.

A ampla aceitação do web service resultou no surgimento de um conjunto de tecnologias suplementares que se tornaram um padrão de fato. Assim ao desenvolver nossos web services devemos considerar o uso de tecnologias que:

- forneça uma descrição de serviço que, no mínimo, consista de um documento WSDL;
- seja capaz de transportar documentos XML utilizando SOAP sobre HTTP.

Estas tecnologias não modificam a funcionalidade do núcleo de um serviço web, tanto como o faz sua habilidade para se representar e comunicar num modo padrão. Muitas das convenções de arquitetura expressadas neste artigo assumem que SOAP e WSDL fazem parte da estrutura de web services descrita.

Além disto, é normal que um Web service seja:

- capaz de agir como o solicitante e o provedor de um serviço;
- registrado com um discovery agent através do qual possam ser localizados.

Numa conversação típica com um web service, o cliente iniciador do pedido é um web service também. Como mostrado na **Figura 2**, qualquer interface exposta por este client service também o qualifica como um serviço a partir do qual outros serviços podem solicitar informação. Posto isto, web services não se encaixam no modelo clássico de cliente-servidor. Na verdade, eles tendem a estabelecer um sistema ponto-a-ponto, onde cada serviço pode atuar como cliente ou servidor.

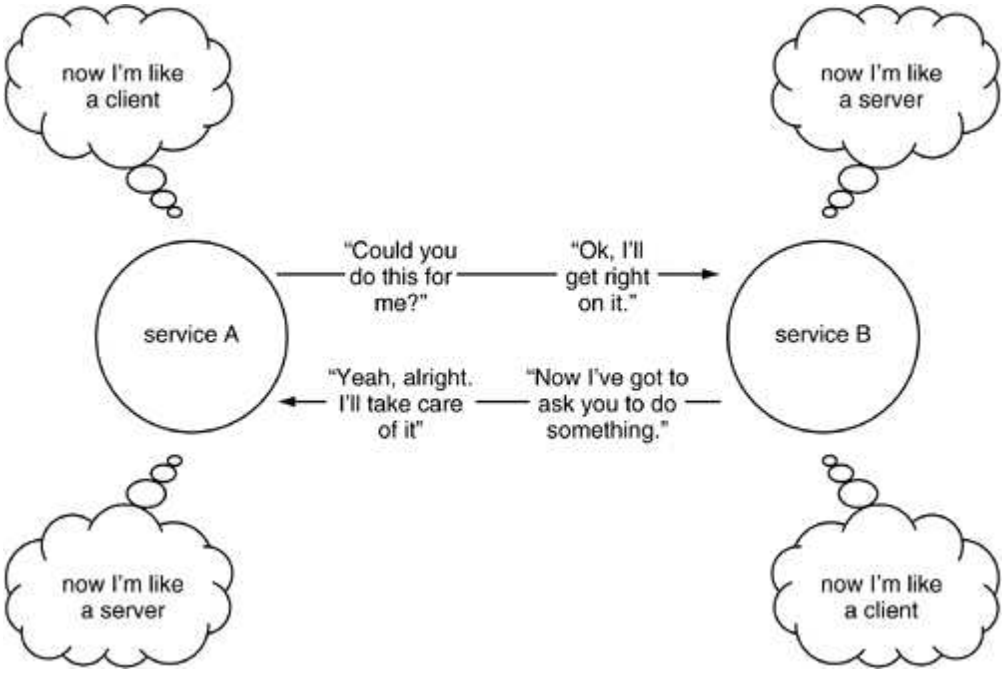

Figura 2: Troca de papéis do Web services durante uma conversação

- Now I´m like a client: agora sou um cliente;
- Now I´m like a server: agora sou um servidor;
- Service: serviços;
- Could you do this for me: poderia fazer isso por mim?;
- Ok, i´ll get right on it: Ok;
- Now I´ve got to ask you to do something: agora eu gostaria de lhe pedir algo;
- Yeah, alright. I´ll take care of if: Tudo bem, é só falar.

Service-oriented architecture (SOA)

Como mencionado anteriormente, adicionar uma aplicação com uns poucos web services não é nenhum problema. Esta integração limitada pode ser apropriada para uma experiência de aprendizado, ou para complementar a arquitetura de uma aplicação existente com uma peça de funcionalidade baseada em serviços que atende a um requisito específico do projeto. No entanto, isto não estabelece uma arquitetura orientada a serviço. Existe uma clara diferença entre:

- uma aplicação que usa web service;
- uma aplicação baseada numa arquitetura orientada a serviços.

Uma SOA é um modelo de projeto com um conceito profundamente amarrado à questão do encapsulamento de aplicação. A arquitetura resultante estabelece essencialmente um paradigma de projeto, no qual web services são os blocos de construção chave. Isto quer dizer que ao migrar a arquitetura da sua aplicação para uma SOA, estabelece-se um compromisso com os princípios de projeto de web services e a tecnologia correspondente, como partes fundamentais do seu ambiente técnico.

Uma SOA baseada em XML web service é construída sobre camadas de tecnologia XML estabelecidas, focada em expor a lógica de aplicação existente como um serviço fracamente acoplado. Para apoiar este modelo, uma SOA promove o uso de um mecanismo de discovery por serviços via um service broker ou discovery agent.

A **Figura 3** mostra como uma SOA altera a arquitetura multicamada existente, ao introduzir uma camada lógica que, através do uso de interfaces programáticas padrão (providas pelo web services), estabelece um ponto comum de integração. Esta camada de integração de serviços constitui a base para um novo modelo que pode se estender além do escopo de uma única aplicação, unificando plataformas legadas díspares em um ambiente aberto. Quando web services são utilizados para integração cruzada de aplicações (ver **Figura 4**), elas se estabelecem como parte da infra-estrutura do sistema.

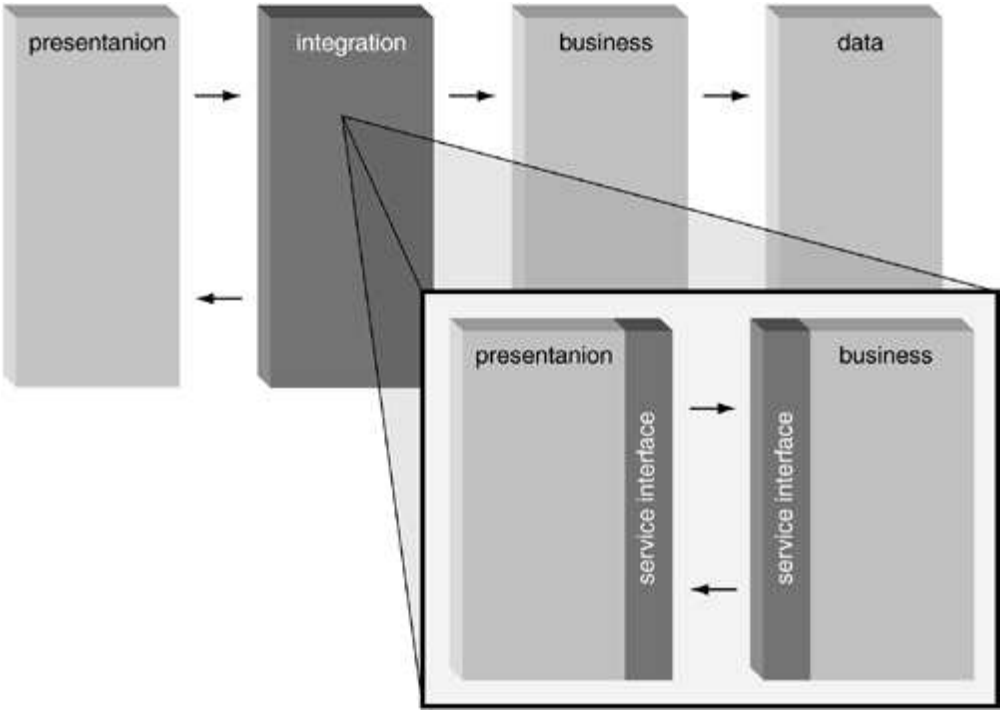

Figura 3: Uma representação lógica de uma arquitetura orientada a serviços

- Presentation: apresentação;
- Integration: integração;
- Business: negócio;
- Data: dado;
- Service interface: interface de serviço.

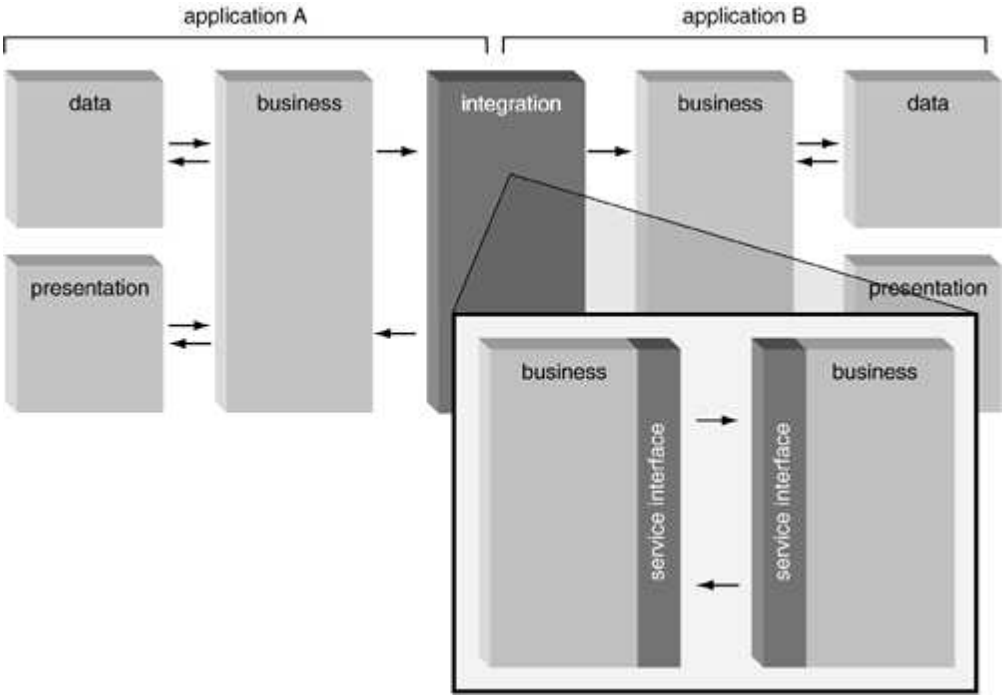

Figura 4: Uma representação lógica de uma arquitetura de integração orientada a serviço

É importante se conscientizar quanto ao acréscimo de complexidade de projeto introduzido pelo SOA. Mais ainda do que em um ambiente n-camada, projetistas de aplicação devem considerar de uma forma completa como a introdução de serviços vai afetar dados existentes e modelos de negócio.

Na medida em que a utilização de serviços se diversifica, o significado dos requisitos de segurança e escalabilidade são amplificados. Ambientes orientados a serviço bem projetados tentarão vencer estes desafios com infra-estrutura adequada, ao invés de utilizar soluções sob medida, específicas de aplicação

Os papéis e cenários ilustrados nas próximas duas seções estão limitados somente ao assunto web service. A estrutura de mensagens SOAP subjacente será explicada em separado, no próximo artigo desta série.

Papéis web service

Serviços podem assumir diferentes papéis quando envolvidos em diversos cenários de interação. Dependendo do contexto pelo qual é visualizado, assim como o estado da tarefa rodando no momento, o mesmo web service pode trocar de papéis ou ser designado para múltiplos papéis simultâneos:

Provedor de serviços

Agindo como um provedor de serviços, um web service expõe uma interface pública através da qual pode ser chamado por solicitantes do serviço. Um provedor de serviços disponibiliza esta interface publicando uma descrição do serviço. Num modelo cliente-servidor, o provedor de serviço pode ser comparado ao servidor.

O termo "provedor de serviço" pode também ser usado para descrever a organização ou ambiente que hospeda (provê) o web service.

Um provedor de serviço pode também agir como um solicitante de serviço. Por exemplo, um web service pode atuar como um provedor de serviço quando um solicitante de serviço lhe pede para executar uma função. Pode então atuar como um solicitante de serviço quando mais tarde contata o solicitante de serviço original (agora agindo como um provedor de serviço) para solicitar informação de status.

Solicitante de serviço

Um solicitante de serviço é o remetente de uma mensagem web service ou o programa de software solicitando um web service específico. O solicitante de serviço é comparável ao cliente dentro de um modelo cliente-servidor padrão. Solicitantes de serviços são às vezes chamados de consumidores de serviços.

Um solicitante de serviço pode também ser um provedor de serviço. Por exemplo, num modelo de solicitação e resposta, o web service iniciador primeiro age como um solicitante de serviço ao requerer informações do provedor de serviço. O mesmo web service então, faz o papel de um provedor de serviço ao responder à solicitação original.

Intermediário

O papel de intermediário é assumido pelo web service quando ele recebe a mensagem de um solicitante de serviço e a passa adiante para o provedor de serviço. Neste caso, ele pode também agir como um provedor de serviço (recebendo a mensagem) e como um solicitante de serviço (passando adiante a mensagem).

Intermediários podem existir em muitas formas diferentes. Alguns são passivos e simplesmente re-transmitem ou roteam as mensagens, enquanto outros processam ativamente uma mensagem antes de repassá-la. Tipicamente, aos intermediários só é permitido o processamento e modificação do cabeçalho da mensagem. Para preservar a integridade da mensagem, seus dados não devem ser alterados.

Remetente inicial

Como o web service responsável por iniciar a transmissão da mensagem, remetentes iniciais também podem ser considerados solicitantes de serviço. Este termo existe para ajudar a diferenciar o primeiro web service que envia uma mensagem, dos intermediários também qualificados como solicitantes de serviço.

Receptor final

O último Web service a receber uma mensagem é o receptor final. Estes serviços representam o destino final de uma mensagem e também podem ser considerados provedores de serviço.

Interação web service

Quando mensagens são passadas entre dois ou mais web services, uma variedade de cenários de interação pode acontecer. A seguir, termos comuns utilizados para identificar e etiquetar estes cenários serão apresentados.

Caminho da mensagem

A rota pela qual a mensagem viaja é o caminho da mensagem. Deve consistir de um remetente inicial e um receptor final e pode conter nenhum, um, ou mais de um intermediários. A **Figura 5** ilustra um caminho de mensagem simples.

O caminho de transmissão atual percorrido por uma mensagem pode ser dinamicamente determinado por roteadores intermediários. A lógica de roteamento pode ser ativada em resposta à carga de requisitos de balanceamento, ou pode ser baseada nas características da mensagem e outras variáveis lidas e processadas pelo intermediário em tempo de execução. A **Figura 6** descreve como uma mensagem é enviada via um ou dois caminhos de mensagem possíveis, tal como é determinado pelo roteador intermediário.

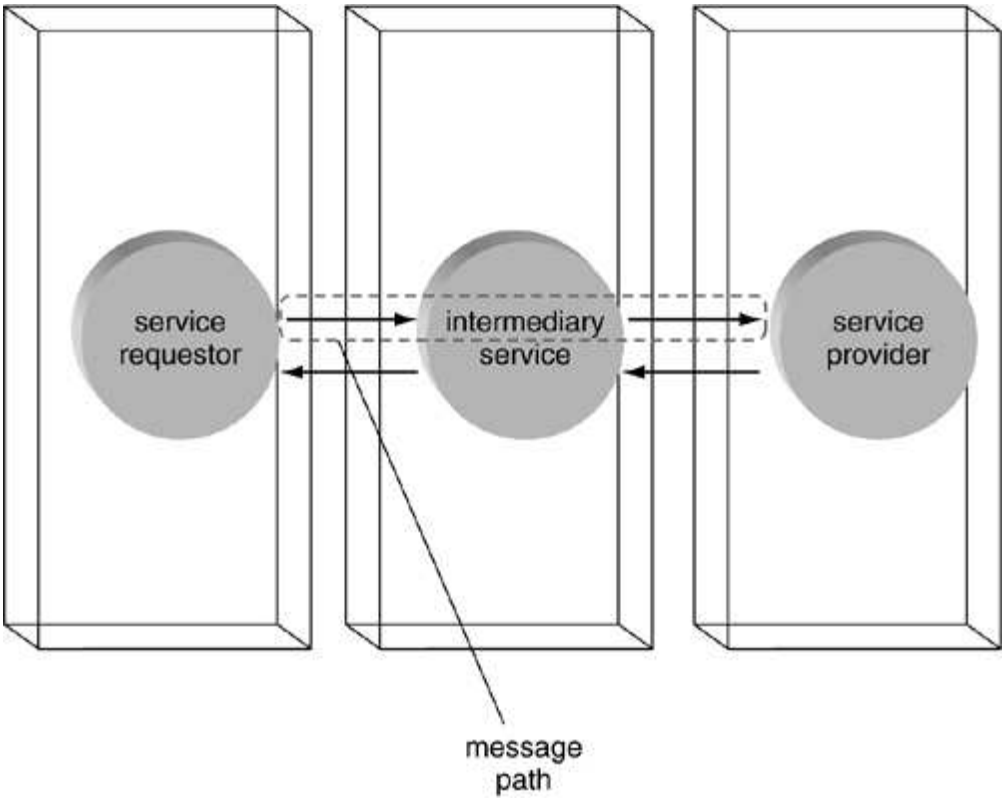

Figura 5: Um caminho de mensagem formado por três Web services

- Message path: caminho da mensagem.

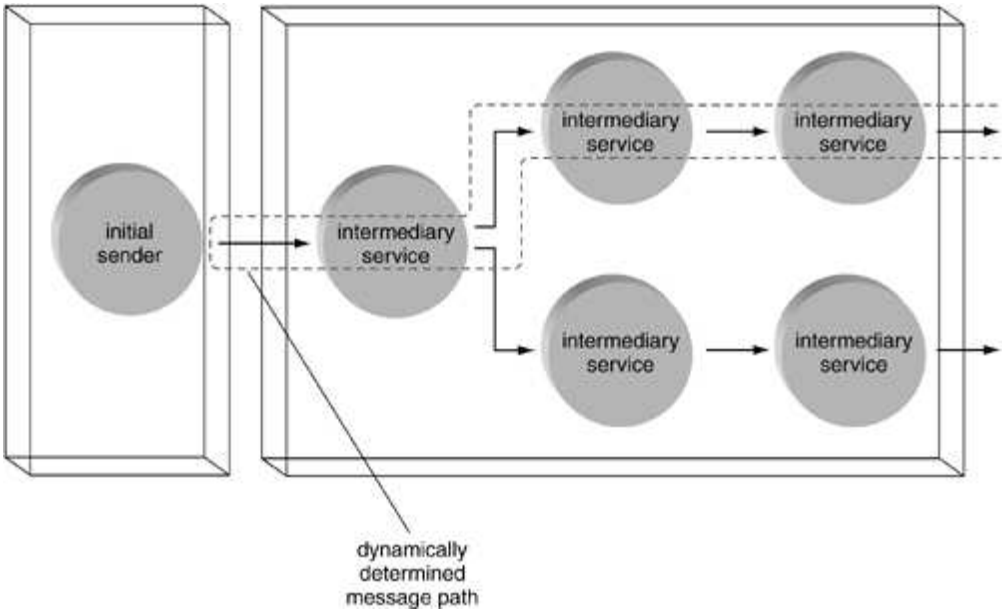

Figura 6: Uma mensagem dinamicamente determinada por um roteador intermediário

- Dynamically determined message path: caminho da mensagem determinada dinamicamente.

Padrão de troca de mensagens

Serviços que interagem em um ambiente orientado a serviços tipicamente se enquadram em determinados padrões de troca de mensagens. Padrões típicos incluem:

- solicite e responda (request and response);
- publique e subscreva (publish and subscribe);
- fire and forget - um para um;
- fire and forget - um para muitos ou difusão;

O padrão pedido e resposta é o mais comum quando se está simulando intercâmbio de dados síncronizados. Os demais padrões são usados principalmente para facilitar transferência de dados assíncrona.

Correlação

Correlação (Correlation) é a técnica utilizada para casar mensagens enviadas através de caminhos de mensagem diferentes. É comumente empregada num padrão de intercâmbio de pedido e resposta de mensagem, onde a mensagem de resposta deve estar associada à mensagem original que iniciou a solicitação. Embutir valores de ID sincronizados dentro de mensagens relacionadas é uma técnica frequentemente utilizada para conseguir correlação.

Coreografia

Regras que governam características de comportamento relacionadas à forma como um grupo de web services interagem podem ser aplicadas como uma coreografia (choreography). Estas regras incluem a seqüência na qual web services podem ser chamados, condições que se aplicam à seqüência que está sendo transportada e o padrão de uso que irá definir os cenários de interação permitidos. O escopo de uma coreografia está tipicamente amarrado ao de uma atividade ou tarefa (ver exemplo na **Figura 7**).

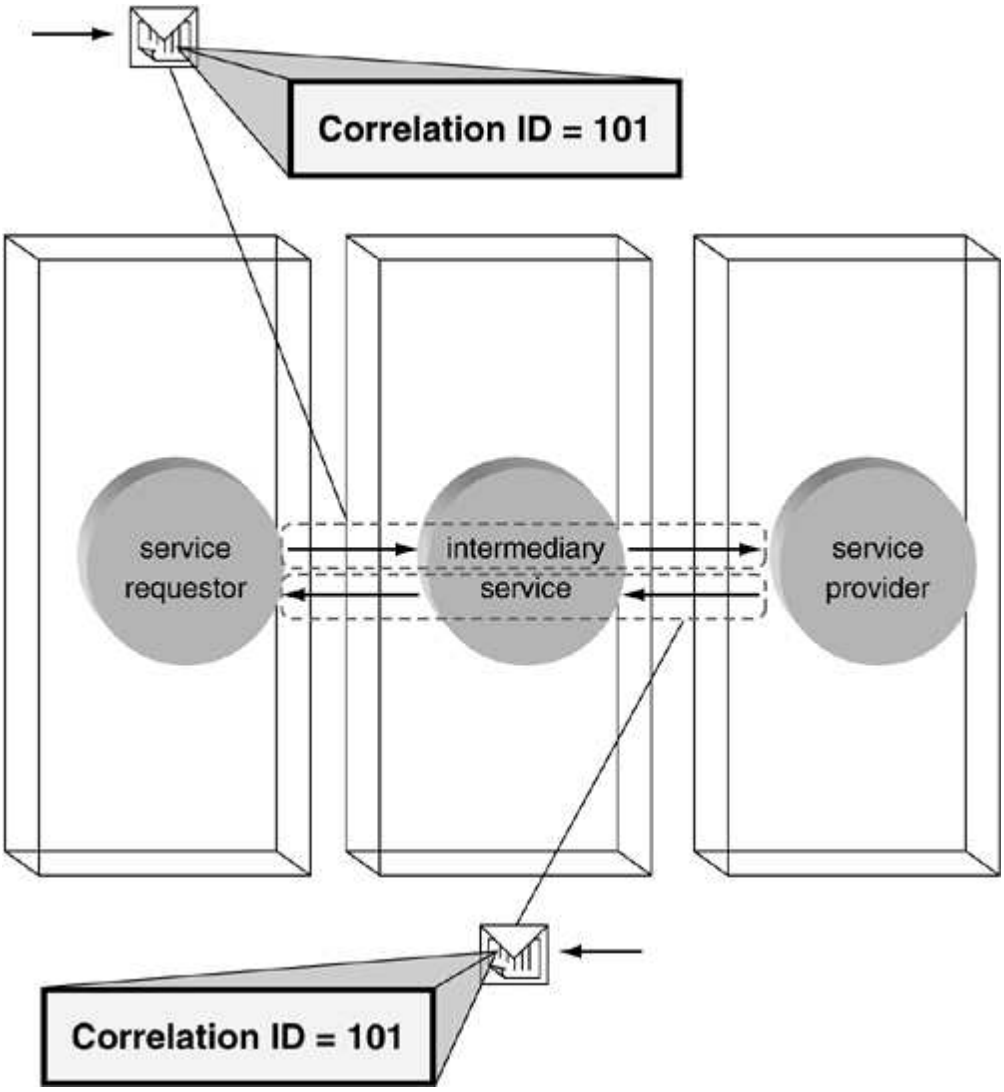

Figura 7: A seqüência de interação de um grupo de serviços sendo comandados por uma coreografia

- Correlation: correlação.

Atividade

Padrões de intercâmbio de mensagens formam a base para atividades de serviços (também conhecidos como tarefas). Uma atividade consiste de um grupo de web services que interagem e colaboram para realizar uma função ou um grupo lógico de funções. A **Figura 8** mostra uma atividade de serviço simples. A diferença entre uma coreografia e uma atividade está no fato de que a atividade é geralmente associada com uma função de aplicação específica, tal como a execução de uma tarefa de negócio.

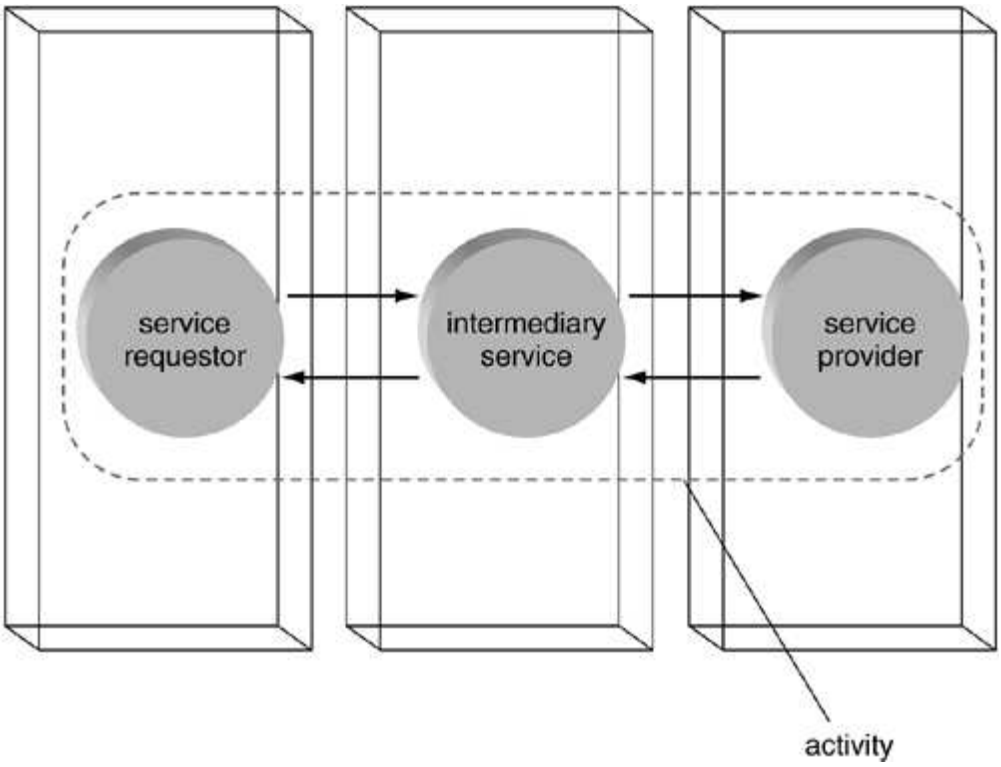

Figura 8: Uma atividade de serviço envolvendo três serviços

- Activity: atividade.

Descrição da estrutura de web services

Um web service é descrito através de uma coleção de documentos de definição. Estes atuam como blocos de construção para uma descrição de serviço:

- Abstrato + Concreto = Definição de Serviço;
- Definição de Serviço + Definições Complementares = Descrição de Serviço.

A **Figura 9** ilustra a relação entre esses documentos.

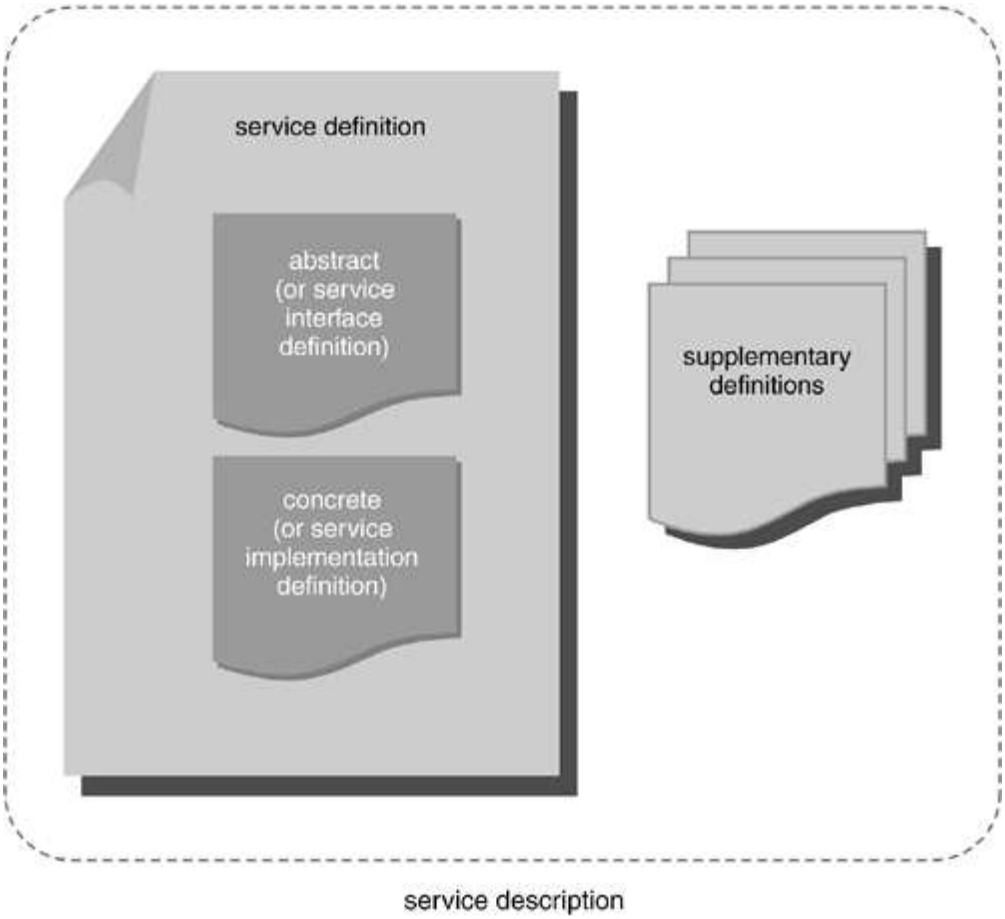

Figura 9: Conteúdo de uma descrição de serviço

- Service description: descrição do serviço;
- Service definifion: definição do serviço;
- Supplementary definitions: definições suplementares;
- Abstract: abstrato;
- Concret: concreto;

- service interface definition: definição da interface do serviço;
- service implementation definition: definição da implementação do serviço.

Abstrato

A descrição de uma interface web service, independente dos detalhes de implementação, é chamada de abstrato (abstract). Descrições deste elemento são fornecidas adiante neste artigo, como parte do tutorial do WSDL.

Concreto

Localização e informação de implementação específicas sobre um web service constituem as partes concretas (concrete) de um documento WSDL. Elas são representadas pelos elementos de ligação (binding), serviço (service) e ponto-de-término (endpoint ou port).

Definição de serviço

Geralmente, o conteúdo de um documento WSDL constitui uma definição de serviço (service definition) que inclui as definições da interface (abstrato) e da implementação (concreto).

Descrição de serviço

Frequentemente, uma descrição de serviço é um único documento WSDL que fornece uma definição de serviço. No entanto, pode também incluir vários documentos de definição adicionais que irão fornecer informações complementares.

Introdução aos web services de primeira geração

A estrutura W3C para web services está fundamentada em três especificações XML fundamentais:

- Linguagem para definição de web service (Web Services Definition Language - WSDL);
- Simple Object Access Protocol (SOAP);
- Universal Description, Discovery, and Integration (UDDI).

Estes padrões de tecnologia, acoplados aos princípios de projeto orientado a serviço, formam um SOA fundamentado na tecnologia XML. Esta arquitetura de web services de primeira geração permite a criação de web services independentes capazes de encapsular unidades isoladas de funcionalidades de negócio. Esta tecnologia tem também algumas limitações, que tem sido contempladas numa segunda geração de especificações. A seção a seguir, fornece um tutorial introdutório para a tecnologia WSDL. SOAP e UDDI serão vistos no próximo artigo da série.

Linguagem para definição de Web Services (WSDL)

Web services devem ser definidos numa forma consistente para que possam ser descobertos e "interfaceados" com outros serviços e aplicações. A WSDL é uma especificação W3C que fornece a linguagem mais avançada para a descrição de definições de web services.

A camada de integração introduzida pela estrutura de web services estabelece um padrão, universalmente reconhecido e com interface programática suportada. Tal como mostrado na **Figura 10**, WSDL permite a comunicação entre essas camadas ao fornecer descrições padronizadas.

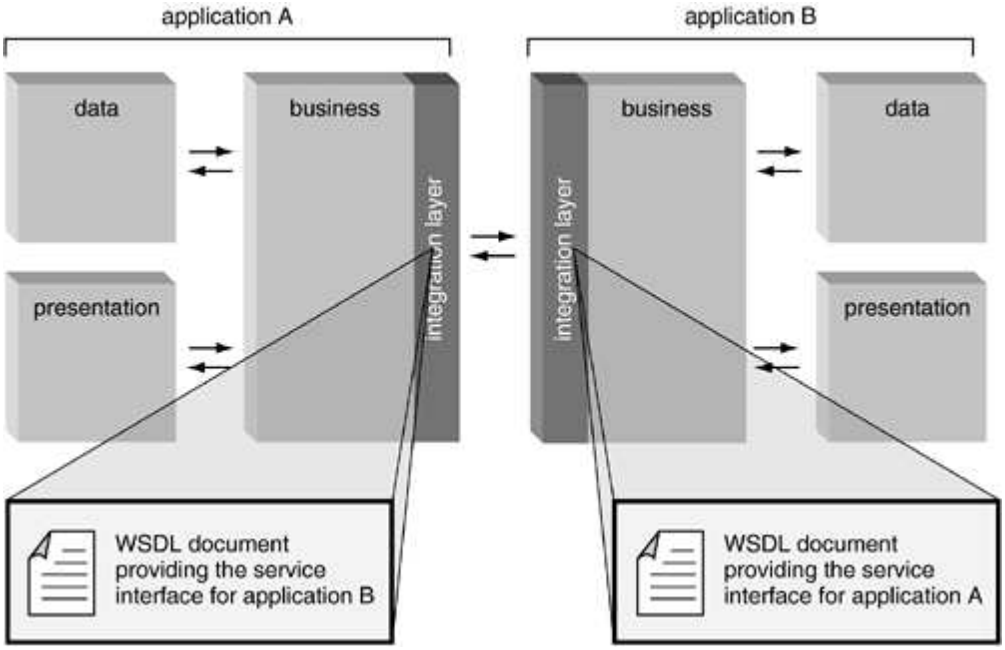

Figura 10: Documentos WSDL representando aplicações web services

- Application: aplicação;
- Integration layer: camada de integração;
- WSDL document providing the service interface for application b: documento WSDLl provendo a interface para o serviço da aplicação b;
- WSDL document providing the service interface for application a: documento WSDL provendo a interface para o serviço da aplicação a.

A melhor forma de entender como é definido um web service e como ele é expresso por um documento WSDL, é caminhar através de cada construtor que coletivamente representa essa definição. Comecemos com o elemento definitions raiz, o qual age como o conteiner para a definição do serviço (ver **Listagem 1**).

Listagem 1: Uma definição de serviço, tal como é expressa pelo construtor definitions

```
<definitions>
        <interface name="Catalog">
        ...
```

```
        </interface>
        <message name="BookInfo">
        ...
        </message>
        <service>
        ...
        </service>
        <binding name="Binding1">
        ...
        </binding>
</definitions>
```

Uma definição WSDL pode conter coleções dos seguintes construtores primários:

- Interface;
- Message;
- Service;
- Binding.

A **Figura 11** ilustra como os primeiros dois construtores representam a definição da interface de serviço e os últimos dois fornecem os detalhes de implementação do serviço.

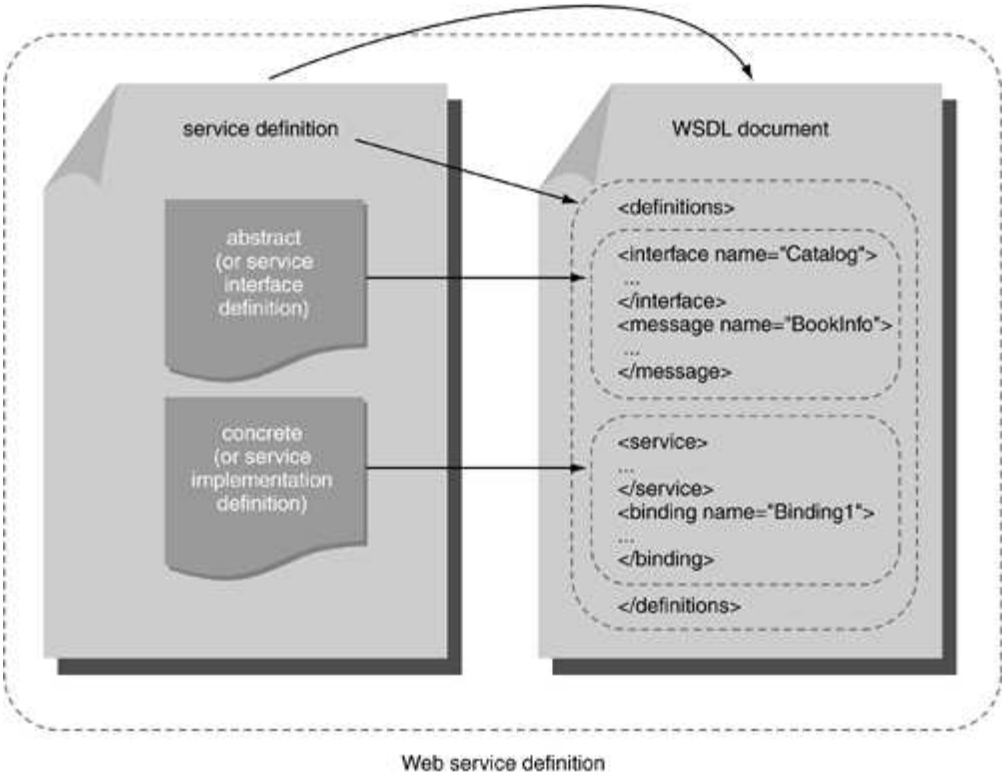

Figura 11: O conteúdo de um documento WSDL, tal como se relaciona com uma definição de serviço

- Web service definition: definição do web service;
- WSDL document: documento WSDL.

Definição de interface abstrata

Interfaces de web services individuais são representadas por elementos interface WSDL. Estes construtores contêm um grupo de operações lógicas correlatas. Numa arquitetura baseada em componentes, um elemento interface WSDL é análogo à interface do componente. Uma operação, portanto, é equivalente a um método de componente, por representar uma única ação ou função. A **Listagem 2** apresenta um exemplo de interface.

Listagem 2: Uma interface representada pelo elemento interface

```
<definitions>
        <interface name="Catalog">
                <operation name="GetBook">
                ...
                </operation>
        </interface>
</definitions>
```

Um elemento operation típico consiste de um grupo de mensagens de entrada e saída correlatas. A execução de uma operation requer a transmissão ou intercâmbio destas mensagens entre o serviço solicitante e o provedor de serviço.

Mensagens operation são representadas por construtores message que são declarados sob os elementos definitions. Os nomes das mensagens são então referenciados nos elementos filho das operation input ou output (ver **Listagem 3**).

Listagem 3: O elemento input dentro do construtor operation referenciando um bloco de mensagem

```
<definitions>
        <message name="BookInfo">
        ...
        </message>
        <interface name="Catalog">
                <operation name="GetBook">
                        <input name="Msg1" message="BookInfo"/>
                </operation>
        </interface>
</definitions>
```

Um elemento message pode conter um ou mais parâmetros input ou output que pertencem a uma operation. Cada elemento part define um destes parâmetros. Ele fornece um conjunto nome/valor (name/value), junto com o tipo de dado associado. Em uma arquitetura baseada em componentes, um part WSDL equivale a um parâmetro de input ou output (ou um valor de retorno) de um método de componente (ver **Listagem 4**).

Listagem 4: Um bloco de mensagem com um construtor part representando parâmetros operation

```
<definitions>
        <message name="BookInfo">
                <part name="title" type="xs:string">
                        Field Guide
                </part>
                <part name="author" type="xs:string">
                        Mr. T
                </part>
        </message>
</definitions>
```

A seguir, um breve resumo dos construtores fundamentais que podem ser montados para estabelecer uma definição de interface abstrata:

- Interfaces: representam interfaces de serviço e podem conter múltiplos operations;
- Operations: representam uma função web service e podem referenciar múltiplas messages;
- Messages: representam uma coleção de parâmetros input ou output e podem conter múltiplas parts;
- Parts: representam parâmetros de dados operation tanto de chegada como de partida.

Definição concrete (implementação)

Sobre os detalhes de implementação, usando os elementos descritos nesta seção, um documento WSDL pode estabelecer detalhes “concrete” de ligação para protocolos, tais como SOAP e HTTP.

Dentro de um documento WSDL, o elemento service representa um ou mais pontos-de-término nos quais o web service pode ser acessado. Estes pontos-de-têrmino consistem de informações de localização e protocolo e são armazenados numa coleção de elementos endpoint (ver **Listagem 5**).

Listagem 5: O elemento ponto-de-término

```
<definitions>
        <service name="Service1">
                <endpoint name="EndPoint1" binding="Binding1">
                ...concrete implementation details...
                </endpoint>
        </service>
</definitions>
```

Agora que descrevemos como um web service pode ser acessado, precisamos definir os requisitos de chamada para cada uma das suas operations. Os elementos binding associam às operations informações de formato de protocolo e mensagem. O construtor operation que reside dentro do bloco binding é semelhante a sua contrapartida na seção interface (ver **Listagem 6**).

Listagem 6: O elemento binding representando uma operation existente

```
<definitions>
        <service name="Service1">
                <binding name="Binding1">
                        <operation>
                                <input name="Msg1" message="book"/>
                        </operation>
                </binding>
```

```
        </service>
</definitions>
```

A descrição de informação concrete dentro de um documento WSDL pode ser resumida assim:

- elementos service: hospedam coleções de ponto-de-término representados individualmente por elementos endpoint;
- elementos endpoint: contem dados endpoint, incluindo endereço físico e informação de protocolo;
- elementos binding: se auto-associam a construtores operation;
- cada endpoint pode referenciar um elemento binding e, portanto, fornecer informação endpoint para a operation subordinada.

Conclusão

Vimos nesse primeiro artigo o princípio de funcionamento dos web services bem como duas de suas tecnologias de base, SOA e WSDL. No próximo artigo, complementaremos o assunto apresentado aqui analisando as tecnologias SOAP e UDDI. Até lá

DevMedia por Thomas Erl
Especialista em aplicações mobile