UNIVERSIDADE FEDERAL DE SANTA CATARINA
CURSO DE SISTEMAS DE INFORMAÇÃO

FELIPE MACHADO ROVARIS

AUTOMATIZAÇÃO NA COTAÇÃO DE LIVROS UTILIZANDO WEB SERVICE.

FLORIANÓPOLIS, JUNHO DE 2007

UNIVERSIDADE FEDERAL DE SANTA CATARINA
CURSO DE SISTEMAS DE INFORMAÇÃO

FELIPE MACHADO ROVARIS

AUTOMATIZAÇÃO NA COTAÇÃO DE LIVROS UTILIZANDO WEB SERVICE.

Trabalho de Conclusão de Curso apresentado ao curso de Sistemas de Informação da Universidade Federal de Santa Catarina – UFSC, como requisito parcial para obtenção do título de Bacharel em Sistemas de Informação.

Orientador(a): Prof. Fernando Cruz

FLORIANÓPOLIS, JUNHO DE 2007

RESUMO

Este trabalho de conclusão de curso tem como principal finalidade a implementação de um sistema para a Sur Distribuidora de Livros, Jornais e Revistas para a automatização dos processos de cotação de livros, usando a tecnologia Web Service, oferecendo aos clientes um acesso fácil e rápido na busca pelos livros. Com esse tipo de serviço, os clientes podem construir seus próprios softwares e consumir os serviços disponibilizados pelo Web Service criado.

Palavras chaves: Web Service - automatização de processos - cotação de livros

ABSTRACT

The present monograph has as main purpose the implementation of a system for Sur Distribuidora de Livros, Jornais e Revistas LTDA for the automatization of the processes of book quotation, using the technology Web Service, offering to the customers an easy and fast access to search for books. With this type of service the customers can construct its softwares and consume the services available from the Web Service created.

Key words: Web Service - automatization of the processes - book quotation

LISTA DE ILUSTRAÇÕES

SUMÁRIO

1 INTRODUÇÃO

Vivemos em um mundo no qual a globalização já se instalou e vem crescendo gradativamente. Com ela, alterações e adaptações no estilo de vida e no cotidiano das pessoas foram ocorrendo. Acompanhando esses novos acontecimentos, as empresas passam a ter de se adaptar às novas mudanças necessárias para manter-se estável no mercado.

Um destaque especial nesse momento de crescimento da globalização deve ser dado à rede mundial de computadores que, impulsionada pela explosão da Internet, faz com que as indústrias, comércios e pessoas se comuniquem e interajam em todo o mundo, unificando-os num só espaço.

Com a popularização da Internet as empresas passaram a se dar conta que havia surgido uma nova maneira de se apresentar junto ao mercado. Tudo se tornara mais acessível. Informações, produtos e comércio que antes exigiam contato físico e movimentação, passaram também a ser apresentados de forma virtual e telefônica.

A partir das oportunidades que a rede mundial de computadores abriu para seus usuários, empresas começaram a trabalhar no desenvolvimento de novidades para satisfazer as necessidades observadas. Dessa forma, softwares de empresas e indústrias que anteriormente eram apenas de uso interno começam a receber alterações e pacotes de versões mais atuais com suporte à Internet, iniciando assim o comércio eletrônico.

Percebendo a possibilidade de utilizar uma nova maneira de se comercializar produtos via Internet, surge a proposta da criação de uma aplicação no ambiente Web para a Sur Distribuidora de Livros, Jornais e Revistas Ltda, pois observa-se que a empresa possui um processo de realização de pedido de livros que pode ser incrementado, possibilitando ao comprador maiores informações do produto que está adquirindo. Além disso, a proposta visa oferecer a esta empresa uma melhor organização interna nas solicitações dos seus consumidores.

Por trazer como grande característica a interoperabilidade, Web Service é a tecnologia a ser utilizada no desenvolvimento da aplicação proposta. Ela proporcionará, automaticamente, maior flexibilidade na solução dos problemas, devido à utilização dos padrões da linguagem XML, possibilitando que no futuro haja facilidade na integração com outros aplicativos.

Para a Sur, a utilização de Web Services fará com que os processos sejam executados com maior agilidade e eficiência, melhorando a qualidade na comunicação entre o setor logístico da empresa e a sua distribuição, devido ao fato de se reduzir a intervenção de alguma pessoa.

1.1 SITUAÇÃO PROBLEMA

A Sur Distribuidora de Livros, Jornais e Revistas Ltda é considerada hoje uma das maiores distribuidoras de publicações da língua espanhola. A empresa é composta por 8 funcionários que atuam no desenvolvimento da mesma.

A empresa atende grandes livrarias de todas as partes do Brasil e os pedidos dos materiais em espanhol são realizados através do envio de fax, e-mail ou telefonema. Após o recebimento dessas solicitações, a Sur realiza o levantamento de estoque, e em seguida faz as cotações dos títulos solicitados. Caso os materiais não estejam disponíveis em estoque, a Sur entra em contato com os seus fornecedores para verificar a disponibilidade do mesmo e sua respectiva cotação, podendo assim dar um retorno ao seu cliente. Isso faz com que o processo de atendimento ao cliente se torne demorado.

A solução deste problema seria a implantação de um Web Service. Como a utilização dessa tecnologia poderia melhorar o atendimento ao cliente? Como agilizaria o processo de compra das livrarias?

1.2 OBJETIVOS

1.2.1 Objetivo Geral

Implantar um Web Service, para melhorar o atendimento ao cliente e padronizar o processo de cotação de livros, trazendo assim uma satisfação maior ao consumidor e facilitando os processos internos da Sur Distribuidora de Livros, Jornais e Revistas Ltda.

1.2.2 Objetivos Específicos

- Realizar um levantamento de todos os métodos utilizados para recebimento de pedidos possuídos pela Sur;
- Adquirir todas as informações básicas dos materiais comercializados;
- Desenvolver uma aplicação que padronize a cotação de livros.

1.3 JUSTIFICATIVA

Atualmente, estamos sempre buscando novos mecanismos para melhorar aquilo que fazemos. Com isso procuramos incessantemente uma forma eficaz de automatizar as tarefas, diminuindo o tempo de execução e também o erro humano. Percebemos então, que um modo prático e eficiente de concretizar essas idéias seria a criação de soluções através de programas de computadores.

Com a grande expansão da Internet no mundo, as informações ficaram muito mais acessíveis. Mensagens que levavam dias para serem entregues a uma determinada pessoa hoje é entregue quase que instantaneamente através de e-mails. Junto a isso, não apenas as correspondências se tornaram mais eficazes, mas de um modo geral tudo ficou mais rápido e de fácil acesso. Notícias, fotos, vídeos, músicas, tudo hoje em dia está disponível na rede mundial de computadores. E por isso as empresas não poderiam ficar de fora. Elas procuram cada vez mais espaço no mundo virtual, que já não é tão novo, mas que oferece inúmeras opções de como solucionar um determinado problema.

Estudos relatam que, atualmente, a maioria das empresas procura a solução de seus problemas em softwares que sejam integrados à web, para que, de um certo modo, informações da empresa sejam acessadas de qualquer parte do mundo, facilitando o acesso tanto para funcionários quanto para clientes.

A implementação de programas sempre foi um ponto chave no desenvolvimento de soluções, sendo importante fazer uma boa escolha na linguagem de programação e também na plataforma que seria feita a execução do software. Outro aspecto relevante é a comunicação do programa, pois muitas vezes o desenvolvedor precisa repetir a solução em outra criação de software pelo fato de programas distintos não poderem se comunicar entre si. Com essa visão, acredita-se que Web Services se tornam uma tecnologia competente no desenvolvimento de softwares.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 A ORIGEM DA XML

Em meados dos anos 90, iniciou-se a partir de uma organização intitulada World Wide Web Consortium (W3C) a criação de uma nova linguagem de marcação na qual a flexibilidade seria uma característica determinante, trazendo consigo a simplicidade da HTML. A insatisfação pela não padronização das atuais linguagens existentes, fez com que surgisse o objetivo inicial de criar um formato simples de ser lido por um software e que pudesse comunicar-se facilmente com outras linguagens. Derivando de um padrão mais amplo conhecido como a SGML (Standard Generalized Markup Language, ou Linguagem Padronizada de Marcação Genérica) – ISSO 8879, que tem sua atuação em padronização de conteúdo técnico, a XML (eXtensible Markup Language) possui como foco principal o compartilhamento de maneira simples dos dados trafegados na Internet. Por isso algumas características encontram-se expostas na XML, tais como:

- **Divisão dos dados da interface com o usuário**: Define-se como a divisão do conteúdo, proveniente de algum lugar, da formatação, interface a ser apresentada ao usuário. Esta divisão possibilita ao desenvolvedor tratar a junção dos dados de diferentes maneiras, e de uma forma organizada. A comunicação XML é apresentada a seguir, na figura 1.

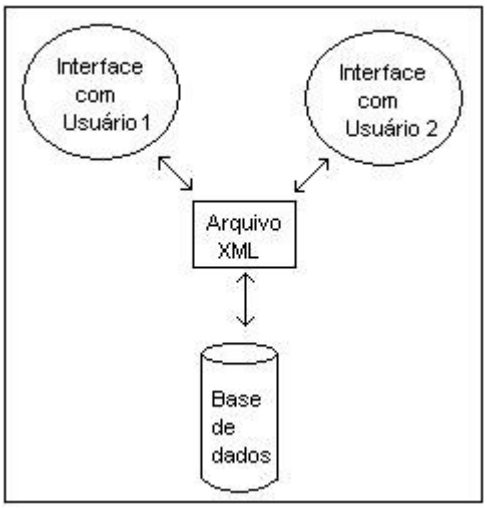

Figura 1: Comunicação XML

- **Estrutura na forma de árvore**: Estrutura de dados bidimensional, segundo DEITEL (2003, pg.1007) “O nodo-raiz é o primeiro nodo em uma árvore. Cada link no nodo-raiz faz referência a um filho”. A figura 2 apresenta a estrutura citada.

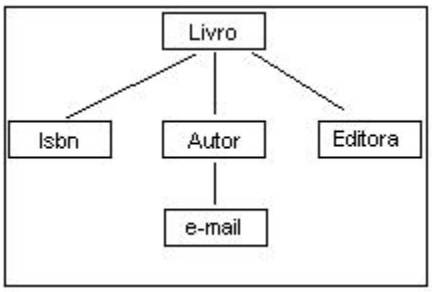

Figura 2: Estrutura em forma de árvore

- **Definição das regras**: Chamada de DTDs (Document Type Definitions), as regras opcionais, que são aplicadas aos dados para verificar sua validação. Através destas é possível construir uma boa estrutura de um documento XML.
- **Tags personalizadas:** É possível criar novas tags sem limites, desde que as mesmas estejam dentro do escopo do problema e respeitem as suas regras e sua estrutura.
- **Comunicação entre bases de dados distintas:** Com a capacidade de estruturar os dados de forma simples é possível resgatar as informações de determinadas bases de dados e realizar uma união.
- **Flexibilidade para aplicações Web**: A estrutura em árvore facilita muito sua flexibilidade, pois os dados podem ser distribuídos para desktops, servidores ou outros locais com enorme facilidade.

2.1.1 Estrutura dos Dados

XML é um conjunto de informação dividido em módulos, que projeta formatos de texto,

permitindo a estruturação dos dados. Todo documento XML descreve a mesma estrutura padrão, sendo chamado de prólogo o cabeçalho, onde é feita a identificação do documento como sendo XML, sua respectiva versão e tipo de documento que está sendo associado, e instância a parte que vem logo abaixo trazendo os dados organizados definidos pela sua hierarquia. XML facilita ao computador ler e escrever dados, e tornar uma estrutura não ambígua e de fácil compreensão ao ser humano.

```
01 <?xml version=”1.0” encoding="ISO-8859-1"?>
02 <lista>
03    <livro id=”1”>
04          <isbn>8420471836</isbn>
05          <titulo>Cien años de soledad Edición Conmemorativa</titulo>
06          <autor>Márquez, Gabriel García</autor>
07          <editora>Alfaguara</editora>
08    </livro>
09    <livro id”2”>
10          <isbn>9502804201</isbn>
11          <titulo>Gladiador</titulo>
12          <autor>Russel, Gordon</autor>
13          <editora>Grijalbo</editora>
14    </livro>
15</lista>
```

- Na linha 01 encontra-se a definição da versão usada do XML e sua codificação.
- Na linha 02 é descrito o elemento raiz do documento.
- No período da linha 03 à linha 14 é apresentado os elementos do documento raiz.
- A linha 15 faz o fechamento da raiz.

É permitido que atributos possam ser inseridos no início da tag dos elementos XML, sendo muitas vezes usados para dispor de informações adicionais, como no exemplo acima demonstrado pelas linhas 03 e 09.

2.1.2 Namespace

Devido à capacidade de XML poder gerar tags relacionadas aos dados utilizados, isso pode causar um transtorno quando mesmas tags são definidas para situações com semânticas diferentes.

Exemplo 01:

```
<lista>
    <autor id=”1”>
        <nome>Márquez, Gabriel García</nome>
        <nacionalidade>Colombiano</nacionalidade>
    </autor>
</lista>
```

Exemplo 02:

```
<lista>
    <editora id=”1”>
        <nome>Sudamericana</nome>
        <pais>Argentina</pais>
    </autor>
</lista>
```

Observa-se que os exemplos acima possuem o mesmo nome do elemento raiz, portanto os dados propriamente ditos no seu corpo são obtidos de informações diferentes. Para contornar esse conflito utiliza-se o conceito de namespace, definido por um prefixo adicional que determina qual contexto está associado, como pode ser observado no exemplo a seguir.

Exemplo 03:

```
<?xml version=”1.0” encoding="ISO-8859-1"?>
<livros:root xmlns:a=”autores” xmlns:e=”editoras”>
```

```
<a:lista>
    <a:autor id=”1”>
        <a : nome>Márquez, Gabriel García</a:nome>
        <a: nacionalidade>Colombiano</a:nacionalidade>
    </a:autor>
</a:lista>
<e:lista>
    <e:editora id=”1”>
        <e:nome>Sudamericana</e:nome>
        <e:pais>Argentina</e:pais>
    </e:autor>
</e:lista>
</livros:root>
```

2.1.3 XML Schema

Segundo a W3C, “os Schemas de XML expressam vocabulários compartilhados e permitem que as máquinas realizem as regras feitas pelas pessoas. Fornece um significado definindo a estrutura, o índice e a semântica dos documentos XML”.

Esta linguagem, diferentemente ao DTD (Document Type Definition), cuja sintaxe não é baseada no formato XML, possui sua base em arquivos escritos em XML que define as regras e cardinalidade dos elementos, valores e atributos válidos para a elaboração dos arquivos XML.

Possui suporte a definição de novos tipos de dados, determinados pelo usuário e também permite a reutilização de código.

Exemplo 01:

```
<xsd:simpleType name=”titulo” type=”xsd:string”/>
```

Aplica-se ao exemplo acima a definição de um tipo simples, com o qual é definido o tipo de dado no formato texto (string).

Para elementos complexos, deve ser criado um novo tipo, usando a chave complexType, que define o nome do novo tipo e seu respectivo conjunto, que no geral são elementos simples.

Exemplo 02:

```
<xsd:complexType name="livro">
    <xsd:sequence>
        <xsd:simpleType name="isbn" type="string"/>
        <xsd:simpleType name="titulo" type="string"/>
        <xsd:simpleType name="autor" type="string"/>
        <xsd:simpleType name="editora" type="string"/>
    </xsd:sequence>
</xsd:complexType>
```

2.2 SOA (ARQUITETURA ORIENTADA A SERVIÇOS)

Atualmente, devido à grande concorrência entre as empresas desenvolvedoras de software, existe uma forte pressão para que as mesmas disponham de comunicação entre seus aplicativos aumentando a agilidade na resposta das condições de mudança de negócios. Sobre essa forte demanda, SOA é a base da infra-estrutura de TI do futuro.

Segundo a SUN, "SOA é um estilo de arquitetura para as aplicações que usam os serviços disponíveis em uma rede. Promove o acoplamento baixo entre componentes de software de modo que possam ser reusados. As aplicações em SOA são construídas baseadas em serviços. Um serviço é uma execução de uma funcionalidade bem definida do negócio, e tais serviços podem então ser consumidos por clientes em aplicações ou em processos diferentes do negócio. SOA permite o reuso dos recursos existentes, onde os serviços novos podem ser criados existindo também na infra-estrutura dos sistemas. Permite o reuso das aplicações existentes, e promete a interoperabilidade entre aplicações e tecnologias heterogêneas. SOA fornece um nível da flexibilidade que antes não era possível".

Uma arquitetura orientada a serviços, SOA, é uma coleção dos serviços que se comunicam um com o outro. Os serviços são contidos em si próprios e não dependem do contexto ou do estado do outro serviço. Trabalham dentro de uma arquitetura distribuída dos sistemas.

No ponto de vista dessa arquitetura, SOA oferece um serviço que é definido como uma função de um sistema que é viabilizado para outro sistema na forma de um serviço. As interfaces desses sistemas devem ser bem esclarecidas para que seja realizado um desempenho independente entre os sistemas.

A principal característica de SOA é fornecer a interoperabilidade de sistemas heterogêneos, buscando com isso a utilização de protocolos padrões para sua comunicação. Os Web Services são a maneira preferida de realizar SOA, mas também pode ser utilizada em qualquer tecnologia que utiliza padrões Web.

SOA ajuda a organizações a agilizar processos de modo que possam fazer o negócio de forma mais eficiente, e adapta-se às necessidades e à competição permitindo o software como um conceito do serviço.

Essa arquitetura é basicamente definida por um simples modelo representado por 3 entidades e é apresentada a seguir, na Figura 3.

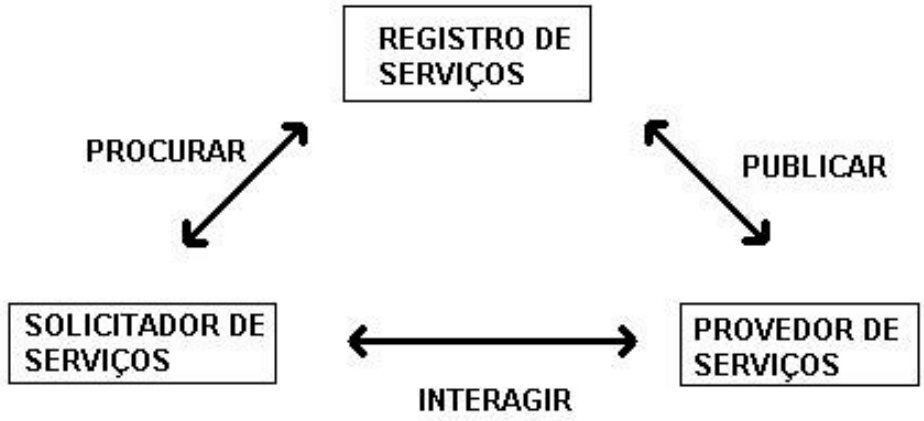

Figura 3: Arquitetura SOA

Registro de serviços: tem como função retornar todas as informações necessárias para a utilização de um serviço.

Provedor de serviços: tem como função, disponibilizar de forma simples os Web Services para que eles possam ser utilizados.

Solicitador de Serviços: é a aplicação que irá consumir os serviços. Por sua vez acessará o registro de serviços para localização dos serviços disponíveis.

Utiliza-se como exemplo uma livraria que deseja realizar a cotação de um determinado livro. O cliente solicita o preço de um do título e por sua vez o Web Service processa e procura na base de dados de uma determinada distribuição. Caso encontre, retorna o valor referente ao título, caso contrário, informa que não foi possível encontrá-lo. Existe também a possibilidade de, no caso de não encontrar esse mesmo método, realizar a chamada de outro serviço web disponibilizado por um outra distribuidora e assim então retornar o preço do livro.

2.3 WEB SERVICES

Web Service é uma tecnologia recente no que diz respeito a padrões de implementação de softwares distribuídos, fazendo com que aplicações consigam compartilhar facilmente informações diferentes. Segundo FERREIRA (2002) “os Web Services são componentes de desenvolvimento de aplicativos, baseados em códigos abertos, que oferecem sistemas e informações pela Internet e que rodam em qualquer dispositivo, de desktops a celulares e handhelds, de forma dinâmica”. Ou seja, Web Services tem como sua principal característica a interoperabilidade, possuindo a capacidade de integrar aplicações e programas distintos, inclusive quando estes mesmos são criados utilizando linguagem ou plataformas diferentes. Segundo Sun (2003), “a principal diferença entre uma aplicação Web e um Web Service é que uma aplicação Web oferece um serviço que exige a intervenção de um usuário, enquanto um Web Service facilita a interação direta entre programa-programa sem intervenção do usuário”. Conforme o IBM Web Services Architecture Team (apud MENDES, 2002, p. 29), Web Services são “aplicações modulares e independentes descritas, distribuídas, publicadas e solicitadas em uma rede TCP/IP, normalmente a Web”.

O que faz com que os componentes de Web Services possuam essa fantástica qualidade de ser interoperáveis é o fato de receberem e enviarem dados utilizando o formato XML. Isso faz com que as aplicações possam ser implementadas em diversas linguagens, desde que essas, no momento do envio ou recebimento dos dados, sejam traduzidas para o padrão XML. Para as empresas, a utilização de Web Services faz com que os processos realizem sua execução com maior agilidade e eficiência, e melhora da qualidade na comunicação entre o setor logístico da empresa e a sua produção. Todo o tipo de comunicação trafegada entre um sistema e outro é muito mais segura devido ao fato de se reduzir a intervenção de alguma pessoa. O medo que as empresas tinham, antigamente, por transitarem seus dados na Internet faz-se reduzir bastante com a utilização de Web Services, pois não são utilizados diretamente os dados da base de dados e sim de arquivos traduzidos em XML.

A identificação dos Web Services geralmente é feita através de URLs, normalmente como é feito numa página Web. O acesso aos mesmos é muito semelhante ao tradicional feito na Internet, o contraste se dá na mensagem que é passada através do cliente e o retorno que é enviado pelo servidor. Essa diferença é causada pelo fato da comunicação trocada entre os dois ser realizada através de mensagens SOAP. O cliente faz uma chamada de requisição passando seus parâmetros necessários, e o servidor faz a execução do método retornando um XML resultante do método invocado. A figura 4 apresenta a comunicação Web Service.

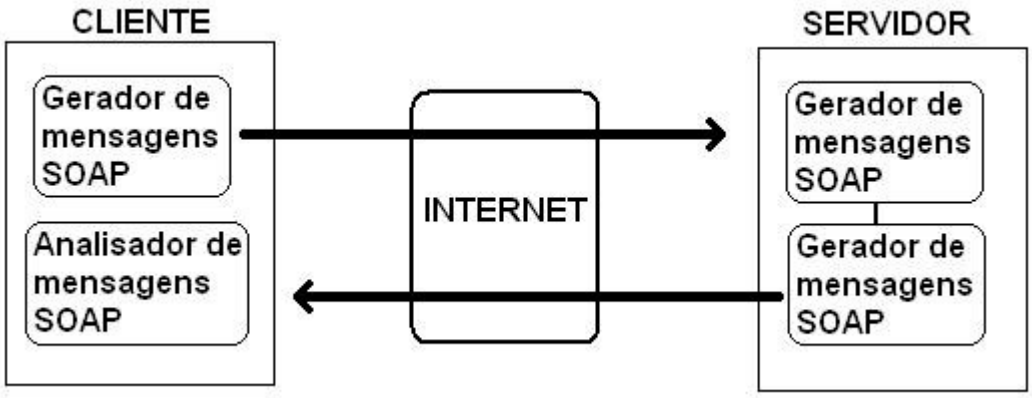

Figura 4: Comunicação Web Service

É um enorme avanço tecnológico essa nova arquitetura representada pelos Web Services,

pelo fato de utilizarem para comunicação protocolos padrões como o HTTP, e a utilização de XML para promover a facilitação da interação dos dados. Isso fez com que as aplicações se tornassem mais transparentes e tivessem interoperabilidade.

De forma geral, os Web Services podem ser implementados por qualquer tipo de linguagem de programação que tenha suporte a sockets.

Web Service torna-se, também, uma forma padrão para implementar arquitetura orientada a serviços (SOA). Essa arquitetura possui o motivo principal do crescimento da tecnologia de Web Services, por possuir sua consistência em coleções de serviços próprios identificados por URLs, juntamente com interfaces definidas em WSDL e manipulação de mensagens XML.

SOA, arquitetura orientada a serviços, diferencia da arquitetura orientada a objetos e da procedural pelo fato que ocorre a interação entre os componentes associados. Normalmente, sistemas orientados a objetos e procedurais são executados através de tipos e nomes. Já a arquitetura orientada a serviços, SOA, realiza comunicação através de troca de mensagens, permitindo que sejam estabelecidas restrições nas informações de envio e recebimento, separando de forma simples a estrutura de descrição da estrutura comportamental.

Segundo Clabby (2002, p.2), “WS pretende afetar a forma como os sistemas de tecnologia de informação são modelados, distribuídos e comprados”. Embora na grande maioria das empresas os sistemas de softwares só se comunicam com outros do mesmo fabricante, Web Services buscam quebrar esse paradigma, trazendo essa tecnologia que surgiu com o intuito de agregar os valores e não fazer com que empresas necessitem mudar de fornecedor, tendo de recomeçar, praticamente.

A figura a seguir mostra como acontece o acesso ao Web Service de uma livraria que, por sua vez, acessa o Web Service de seus fornecedores.

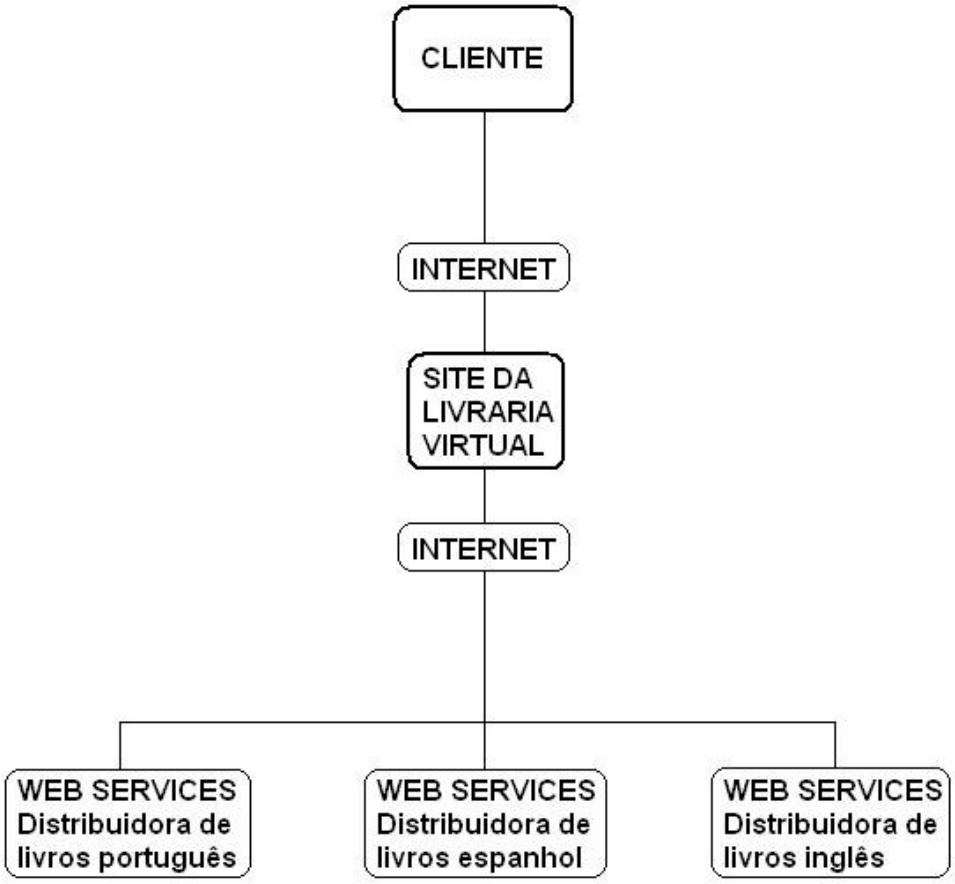

Figura 5: Acesso aos Web Services

2.3.1 Vantagens na Implantação de Web Services

O sucesso desta tecnologia se dá pelo fato de que vários pontos que seriam de grande dificuldade de concretização pelo motivo da grande complexidade e dos custos de implantação, tornam-se quase sempre possíveis, com facilidade e rapidez, conseguindo o seguinte:

Integração de sistemas legados com baixo custo: "Geralmente, os sistemas legados são antigas aplicações de software existentes nas organizações, que são vitais para o funcionamento do negócio" (Siqueira, 2002). Segundo (Brode, 2003), a utilização de sistemas legados se apresenta difícil para adaptação da infra-estrutura de TI à nova demanda do negócio. Portanto, a tecnologia de Web Service, traz consigo a possibilidade de incrementar esses sistemas através da adição de novas funcionalidades, sem que esses mesmos sejam colocados de lado e também gastando

tempo na alteração de código de difícil manuseio. A adoção de Web Service faz com que a empresa seja mais adaptável e reduz significativamente o custo do desenvolvimento e da manutenção do projeto.

Diminuição de custos operacionais: Com certeza a influência na adoção da tecnologia de Web Services faz com que empresas possam apresentar de maneira mais fácil, dinâmica e eficiente, antigos e novos serviços num canal que interliga diversos acessos de forma mais barata e simples. Sem a utilização dessa tecnologia seria enfrentada uma grande complexidade na realização de determinadas tarefas e um tempo de execução também considerável. Um exemplo disso seria a criação de um catálogo de produtos de uma livraria, que permitiria que seus clientes tivessem a oportunidade de acessar tais informações fora do escritório da livraria.

Diminuição no tempo de construção do projeto: Já faz tempo, desde o início da programação orientada a objetos, que o conceito da reutilização de código vem sendo abordado como um ponto importante no desenvolvimento de sistemas e também de fabricantes que procuram reduzir tempo e dinheiro não começando do zero. Também nesse caminho que a abordagem SOA, arquitetura orientada a serviços esta andando. Um objetivo de grande importância é fazer a redução dos custos de desenvolvimento de softwares para que TI consiga obter melhor condições financeiras para a criação de aplicativos novos. Sincronizando os processos com a tecnologia, as empresas criadoras de softwares pretendem aproveitar cada vez mais a reutilização de código já escrito, fazendo com que diminuam o custo e o tempo de geração.

Aproximação dos clientes: Confiança e praticidade são as chaves principais para uma empresa prestadora de serviços. Oferecer isso ao cliente de forma digital torna-se também possível quando utilizada a tecnologia Web Service, pois como são trocadas mensagens XML, podem ser oferecidos aos clientes sistemas com grande interação, personalização e tamanha agilidade nos processos.

Busca de novos horizontes: As empresas vêm buscando de uma forma ou de outra satisfazer seus clientes, oferecendo cada vez mais diversidade de produtos, mas isso envolve grande estrutura. Entretanto, as empresas podem criar pequenos serviços especializados que podem ser

oferecidos separadamente como um sistema, utilizando a infra-estrutura oferecida pela Internet. Também seria ótima opção para os clientes que comprariam esses Web Services, pois com essa nova opção de compra não seria mais preciso adquirir um produto com alto custo financeiro sem o retorno esperado, e sim adquirir aquele serviço específico para resolver um determinado problema, com a possibilidade de mais escolhas devido ao custo ser muito mais baixo. Isso seria de grande interesse para empresas que estariam iniciando no mercado e não teriam muitos recursos de capital para investir em sistemas de grande porte, oferecendo a possibilidade de crescerem gradativamente.

2.3.2 Desvantagens na Utilização de Web Services

Quando se escolhe uma tecnologia a se trabalhar é de fundamental importância saber em qual caso ela será implantada e quais resultados são esperados através da sua utilização. Apesar de serem apresentadas, anteriormente, algumas vantagens na utilização da tecnologia Web Service, essa, como todas as outras já existentes, também dispõe de seus pontos negativos. É importante ressaltar a necessidade de se conhecer os pontos favoráveis e desfavoráveis de uma tecnologia para que ela tenha um melhor desempenho quando implantada na solução do problema.

O ponto principal da utilização de Web Services é estar disponível através da Internet. Isso se torna uma desvantagem a partir do ponto em que tratamos da disponibilidade do serviço. Sabe-se que, mesmo que tudo esteja implementado corretamente, existe a possibilidade de uma URL não estar funcionando corretamente, acarretando assim na indisponibilidade para um negócio ou transação.

Também por depender do tráfego de informações através do uso da Internet, deixa-se muito a desejar o desempenho no processamento dos dados. Já que a taxa de transmissão da Internet é variável, para empresas que necessitam de alto desempenho, este fator pode ser um ponto crucial já que fica sob controle da Internet a comunicação dos dados.

Quando é utilizado um serviço disponibilizado na Web, o cliente que acessa através do seu browser, não tem garantia que o Web Service irá completar determinadas transações com dependências de outras. Estuda-se melhoramentos no que diz respeito a controle de transação e mecanismos de falhas e recuperação.

Como descrito anteriormente, é de extrema importância conhecer bem a tecnologia que será utilizada, pois assim é possível identificar melhor qual possuirá melhor característica para solucionar o problema surgido.

A apresentação de pontos negativos não vem tornar o impedimento da utilização de alguma tecnologia, mas sim ter conhecimento de algumas dificuldades que poderão ser encontradas durante o percurso de implantação.

2.4 SOAP

Segundo o W3C, “SOAP é fundamentalmente sem estado, paradigma de sentido único na troca da mensagem, mas as aplicações podem criar uns testes padrões mais complexos da interação (por exemplo, pedido/resposta, pedido/respostas múltiplas, etc.) combinando tais trocas do único caminho com as características fornecidas por um protocolo subjacente e/ou por uma informação específica da aplicação. O SOAP fornece um mecanismo simples e de pouco peso para a informação estruturada e tipada trocar entre pares em um ambiente descentralizado, distribuído usando XML”.

SOAP, é um protocolo que tem como sua definição realizar a troca de informações de forma modularizada, possuindo XML como base das tecnologias usadas. Sua especificação permite que suas mensagens sejam trafegadas por diversos protocolos, mas geralmente são transmitidas pelo protocolo HTTP. A característica principal de SOAP é a simplicidade e sua capacidade de extensão.

Por trocar mensagens usando os protocolos padrão da Internet, torna-se fácil a passagem por firewalls e outros sistemas de segurança, transparecendo melhor e possibilitando grande flexibilidade. O fluxo de acesso ao Web Service é apresentado a seguir, na Figura 6.

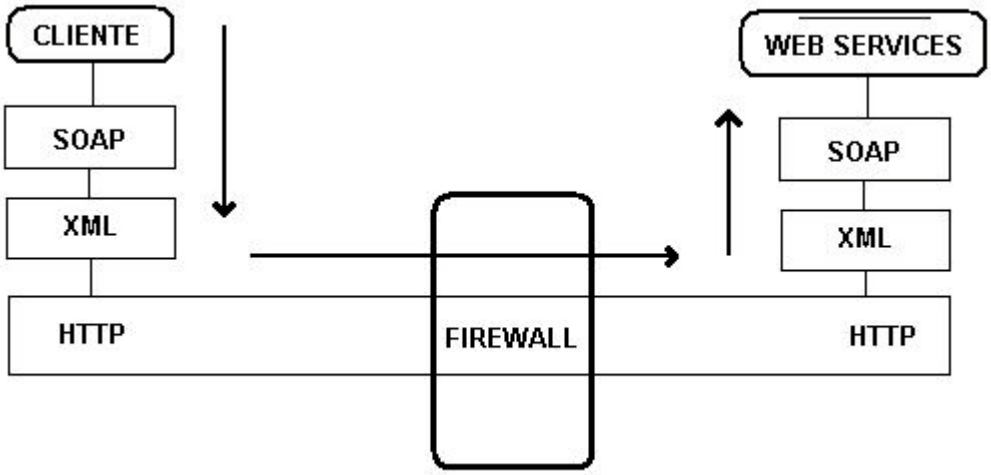

Figura 6: Fluxo de acesso ao Web Service

2.4.1 Estrutura SOAP

A estrutura SOAP é semelhante à descrição de um arquivo XML. Possui uma estrutura básica (Figura 6) dividida em 3 elementos:

- Envelope
- Header (cabeçalho)
- Body (corpo)

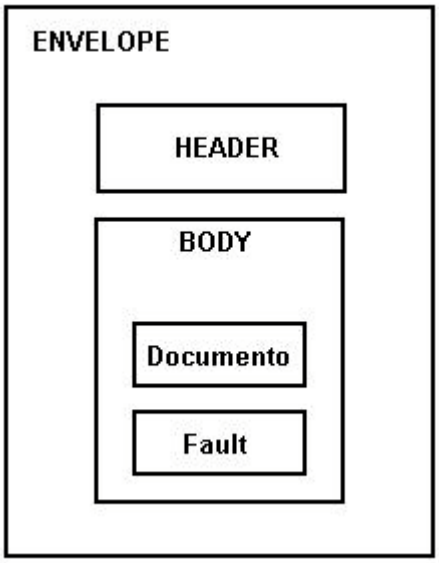

Figura 7: Estrutura SOAP

Envelope: Toda mensagem SOAP o necessita. É o elemento raiz da mensagem XML. O Envelope pode ser preenchido com declarações de namespaces e atributos, definindo o estilo de codificação apresentado no documento XML. A Figura 8 demonstra a estrutura do Envelope.

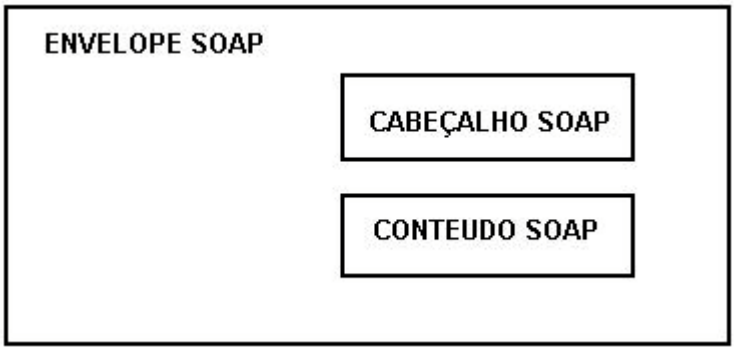

Figura 8: Estrutura Envelope

A seguir, encontra-se um exemplo de código utilizando a tag que define o Envelope.

```
<?xml version="1.0"?>
<soap:Envelope
        xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
```

```
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
  ...

  ...
</soap:Envelope>
```

Header: Diferente do elemento Envelope que deve conter na estrutura SOAP, Header é um cabeçalho opcional. Ele traz informações adicionais específicas. Quando usado, esse elemento deve ser o primeiro elemento do Envelope. São definidos 3 atributos no namespace padrão: actor, mustUnderstand e encodingStyle.

- **Actor**: atributo utilizado para especificar um endereço de ponto final. Abaixo segue exemplo no qual se utiliza uma tag com atributo actor.

```
<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
<soap:Header>
<m:Trans xmlns:m="http://www.w3schools.com/transaction/"
soap:actor="http://www.w3schools.com/appml/">
234
</m:Trans>
</soap:Header>
...
...
</soap:Envelope>
```

- **mustUnderstand**: Semelhante a um atributo booleano, indicando se a entrada do cabeçalho é imperativa ou opcional para que seja feita o processamento. Abaixo segue exemplo no qual se utiliza uma tag com atributo mustUnderstand.

```
<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
<soap:Header>
<m:Trans
xmlns:m="http://www.w3schools.com/transaction/"
soap:mustUnderstand="1">
234
</m:Trans>
</soap:Header>
...
...
</soap:Envelope>
```

- **encodingStyle**: Utilizado para especificar o tipo de codificação de dados utilizado no documento.

Body: Esse elemento não é opcional, é obrigatório. Contém o conteúdo a ser carregado para seu destino final. O elemento Body possui um elemento opcional chamado de Fault, que tem como função transportar as mensagens de erro e status retornadas. Abaixo segue exemplo no qual se utiliza o elemento body.

```
<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
```

```
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
<soap:Body>
  <m:GetPriceResponse xmlns:m="http://www.w3schools.com/prices">
    <m:Price>1.90</m:Price>
  </m:GetPriceResponse>
</soap:Body>
</soap:Envelope>
```

2.4.2 Detecção de Erros Soap

É importante que durante a execução do protocolo SOAP seja feito o registro das informações que foram geradas a partir de algum problema surgido. Para isso usa-se a tag FAULT para controle e registro dos erros. É o objetivo desta seção mostrar uma explanação completa e detalhada das falhas do SOAP de modo que você possa assegurar apropriadamente seus Web Services. Seu retorno ao cliente é imediato, logo que algum erro é apresentado. Não é obrigatória a inserção dessa tag no código SOAP, mas ela é utilizada devido ao interesse em saber as mensagens de respostas. Apresenta-se a estrutura da tag FAULT divida em 4 elementos padrões:

- Faultcode (client, Server, versionMismatch, mustUnderstand)
- Faultstring
- Faultactor
- Detail

Faultcode: Como o nome sugere, este elemento, obrigatório quando utilizado a tag FAULT, representa o código da falha ocorrida. Ele possui quatros especificações diferentes de erros que são definidas:

- **Cliente**: Quando o receptor não consegue processar a mensagem devido a um erro provocado pelo remetente.

- **Server**: Apresenta a falha quando o servidor não consegue executar alguma ação para resolver a mensagem solicitada pelo cliente.

- **versionMismatch**: Basicamente indica o erro, quando não são reconhecidas a versões apresentadas pelo namespace do SOAP.
- **mustUnderstand**: Gera-se um erro quando elementos obrigatórios no cabeçalho não são compreendidos pelo sistema.

Faultstring: Elemento compreensível para identificar a informação ocorrida no faultcode.

Faultactor: Elemento opcional para identificar quem causou a falha

Detail: Retorna informações detalhadas de como estava o servidor na hora que a falha ocorreu.

Demonstra-se, a seguir, a estrutura da XML que representa a estrutura da tag Fault.

```
<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope
 xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
mlns:mh="http://www.servidor.com/ws" >
  <soap:Body>
    <soap:Fault>
        <faultcode>soap:Client</faultcode>
        <faultstring>Isbn contém elemento inválido</faultstring>
        <faultactor>http://www.server.com</faultactor>
        <detail>
                <mh:InvalidIsbnFaultDetail>
                        <offending-value>123456789-D</offending-value>
                        <conformance-rules>
                Os primeiros nove dígitos devem ser numéricos.
                        </conformance-rules>
```

```
            </mh:InvalidIsbnFaultDetail>
        </detail>
    </soap:Fault>
    </soap:Body>
    </soap:Envelope>
```

2.5 WSDL (WEB SERVICES DESCRIPTION LANGUAGE)

Segundo o W3C, “Enquanto os protocolos de comunicação e os formatos da mensagem são padrões na comunidade de Web Services, torna-se cada vez mais possível e importante descrever as comunicações em alguma maneira estruturada. WSDL é um formato de XML para descrever serviços de rede como as informações necessárias para ser feita a conexão com os Web Services. As operações e as mensagens são descritas de forma abstrata, e se limitam a um protocolo de rede e a um formato concreto de mensagem para definir um ponto final. As definições do serviço de WSDL fornecem a documentação para sistemas distribuídos e servem como uma receita para automatizar os detalhes envolvidos em uma comunicação das aplicações.”

Sempre que um cliente pretender saber as funcionalidades de um determinado Web Service, ele terá que acessar a WSDL para entender seu funcionamento. Pois nele terá descrito todas as funcionalidades, tipos de dados, operações e protocolos de ligações necessários para sua utilização.

2.5.1 Estrutura de um WSDL

Uma WSDL usa os seguintes elementos na definição de seus serviços à rede:

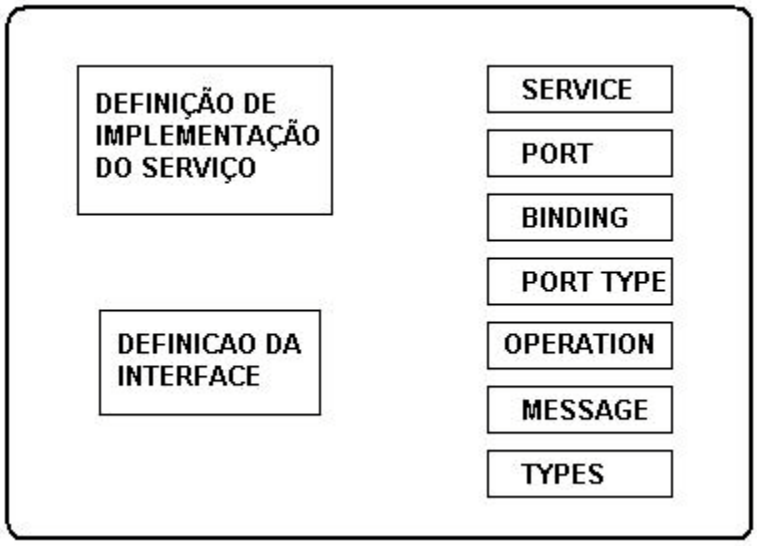

Figura 9: Estrutura WSDL

Types: A tag <types> define os tipos de dados que são aceitos para que as mensagens sejam trocadas. Conforme o W3C, é utilizado "para a neutralidade máxima da interoperabilidade e da plataforma, WSDL prefere o uso de XSD como o tipo canônico do sistema, e trata-o como o tipo intrínseco do sistema". Abaixo segue exemplo no qual se utiliza a tag types.

```
<definitions .... >
  <types>
    <xsd:schema .... />*
  </types>
</definitions>
```

Message: Este elemento consiste em definir de forma abstrata uma ou mais porções de dados que estão sendo transmitidos através da mensagem tipando o atributo. WSDL define diversos atributos tipando-os para uso com XSD. Um exemplo utilizando a tag message é apresentado a seguir.

```
<definitions .... >
  <message name="nmtoken"> *
    <part name="nmtoken" element="qname"? type="qname"?/> *
```

```
    </message>
</definitions>
```

Operation: A tag <operation> define, também de forma abstrata, como uma chamada de método remoto deve mostrar sua estrutura. Observe abaixo um exemplo de sua utilização.

```
<wsdl:operation name="nmtoken" parameterOrder="nmtokens">
        <wsdl:output name="nmtoken"? message="qname"/>
        <wsdl:input name="nmtoken"? message="qname"/>
        <wsdl:fault name="nmtoken" message="qname"/>*
</wsdl:operation>
```

Podem existir 3 tipos de mensagens:

- Input: mensagens de entrada;
- Output: mensagens de saída;
- Fault: Mensagens de falhas;

Port Type: Esse elemento é responsável por juntar as operações abstratas com as mensagens abstratas envolvidas. Segundo o W3C, ele "fornece um nome original entre todos os tipos portuários definidos dentro no original incluindo de WSDL". Abaixo segue exemplo com a sua utilização.

```
<wsdl:definitions .... >
   <wsdl:portType name="nmtoken">
      <wsdl:operation name="nmtoken" .... /> *
   </wsdl:portType>
</wsdl:definitions>
```

Binding: Este elemento é definido segundo o W3C como “o formato da mensagem e o detalhe do protocolo para mapeamento dos elementos operation para cada portType. Os elementos obrigatórios do extensibilidade são usados para especificar a gramática concreta para a entrada (3), a saída (4), e as mensagens de falha (5). A informação obrigatória da ligação (2) e a informação ligada (1) podem também ser especificadas”. Veja um exemplo com sua utilização:

```
<wsdl:definitions .... >
  <wsdl:binding name="nmtoken" type="qname"> *
    <-- extensibility element (1) --> *
    <wsdl:operation name="nmtoken"> *
      <-- extensibility element (2) --> *
      <wsdl:input name="nmtoken"? > ?
        <-- extensibility element (3) -->
      </wsdl:input>
      <wsdl:output name="nmtoken"? > ?
        <-- extensibility element (4) --> *
      </wsdl:output>
      <wsdl:fault name="nmtoken"> *
        <-- extensibility element (5) --> *
      </wsdl:fault>
    </wsdl:operation>
  </wsdl:binding>
</wsdl:definitions>
```

Ports: Este elemento define um ponto final especificando um único endereço para um binding que será utilizado para a comunicação do Web Service. A seguir é apresentado um exemplo de sua utilização.

```
<wsdl:definitions .... >
  <wsdl:service .... > *
```

```
      <wsdl:port name="nmtoken" binding="qname"> *
        <-- extensibility element (1) -->
      </wsdl:port>
    </wsdl:service>
</wsdl:definitions>
```

Service: Este elemento basicamente é um grupo com todas as portas do WSDL. Abaixo segue exemplo no qual se utiliza este atributo.

```
<wsdl:definitions .... >
   <wsdl:service name="nmtoken"> *
      <wsdl:port .... />*
   </wsdl:service>
</wsdl:definitions>
```

2.6 UDDI (UNIVERSAL DISCOVERY, DESCRIPTION AND INTEGRATION)

Segunda a definição da própria Universal Discovery, Description and Integration (UDDI), "é um dos blocos de construção principais requeridos para serviços bem sucedidos dos Web Services. UDDI cria uma plataforma, interoperável, padrão que permite companhias e aplicações, rapidamente, facilmente, e dinamicamente utilizar serviços dos Web Services achados na Internet. UDDI permite também que os registros operacionais sejam mantidos para finalidades diferentes em contextos diferentes. UDDI é um esforço da indústria dirigido por fornecedores principais da plataforma e de software, bem como operadores de marketing e líderes de e-business dentro do consórcio dos padrões do OASIS".

UDDI é como se fosse um diretório online que dá possibilidade de negócios às organizações de uma maneira uniforme para descrição dos seus serviços. Com isso existe também a possibildade de descobrir outros serviços das companhias e compreender os métodos requeridos para conduzir o negócio com uma companhia específica.

Basicamente UDDI é o lugar central para divulgação e encontro dos Web Services criados pelas empresas. Tem como finalidade facilitar a localização desses serviços e propõe como características ser um mecanismo aberto, transparente e padronizado para a descrição dos Web Services; apresentar de forma simplificada os métodos desses servfiços e especificar um acesso central para utilização desses mesmos prestados.

Um registo do negócio de UDDI consiste em três componentes:

- Páginas brancas
- Páginas amarelas
- Páginas verdes

Páginas brancas: é onde ficam especificadas as informações relativas ao negócio. Contém elementos como o nome da companhia, do negócio, contato, identificadores únicos da empresa.

Páginas amarelas: contém informações que descrevem os serviços publicados. Deste modo os web services são classificados em determinadas categorias de acordo com o seu ramo.

Páginas verdes: local no qual são armazenadas as informações técnicas que descrevem o comportamento e funcionalidades dos serviços. Esta informação inclui dados como o local onde se encontra o web service, possibilitando melhores condições para a implementação da aplicação pelos desenvolvedores.

3 ESTUDO DE CASO

Analisando a importância da Sur possuir um mecanismo de cotação de livros mais completo e com maior eficiência na qualidade dos serviços prestados para as livrarias passa-se a buscar novas soluções. Após estudar e verificar os pontos chave que a aplicação necessitará, conclui-se que em termos de implantação, devido à tecnologia ser baseada em padrões abertos, utilizando XML, e capaz de realizar com facilidade a junção com outras aplicações à implementação de Web Service seria uma ótima estratégia para a padronização dos pedidos de livros.

3.1 LEVANTAMENTO DE REQUISITOS

Após ser definido o que se deseja desenvolver, é de extrema importância que seja realizado um levantamento de todas as tecnologias, linguagens e bases de dados que estão sendo utilizadas atualmente, pois é fundamental que essas mesmas sejam possíveis de se agregarem aos novos mecanismos que implantarão no novo projeto.

Além de conhecer as tecnologias já utilizadas, é importante que você tenha um conhecimento dominante, também, da nova tecnologia que será agregada ao sistema. Pois assim pode-se focar melhor no domínio do problema e apresentar melhores resultados.

Entender todos os dados que a sua aplicação fornece é a premissa básica para saber como eles influenciam no negócio. Não é possível tratar algo que não seja compreensível. O entendimento do fluxo de execução deve estar claro desde o início.

No desenvolvimento específico dessa aplicação que será apresentada, percebe-se que já existe uma base de dados criada e o acesso à mesma é realizado através de um linguagem muito conhecida, o PHP.

3.2 OBJETIVO E FLUXO DE EXECUÇÃO DO WEB SERVICE

A aplicação que será criada terá como objetivo apresentar informações relevantes em uma cotação de livros feita por um determinado cliente. A mesma tecnologia utilizada a partir de Web

Services, tomando como base a arquitetura SOA, atuará de forma híbrida, funcionando da seguinte maneira:

1. O cliente acessa o método disponível pelo Web Service.
2. Esse por sua vez realizará a consulta na sua própria base de dados.
3. Serão retornadas as informações dos livros caso exista no banco de dados pesquisado e vazio caso não seja encontrado.
4. Caso sejam encontradas, as informações serão retornadas como resultados pré-definidos.
5. Caso seja retornado o valor vazio na pesquisa feita na sua própria base de dados, o Web Service passa a atuar como cliente.
6. Esse Web Service atuando de forma cliente, acessa outro Web Service, de uma terceira distribuidora, e retorna as informações referentes à base de dados dessa terceira distribuidora.

A Figura 10 apresenta um modelo do fluxo de execução quando acionado o Web Service.

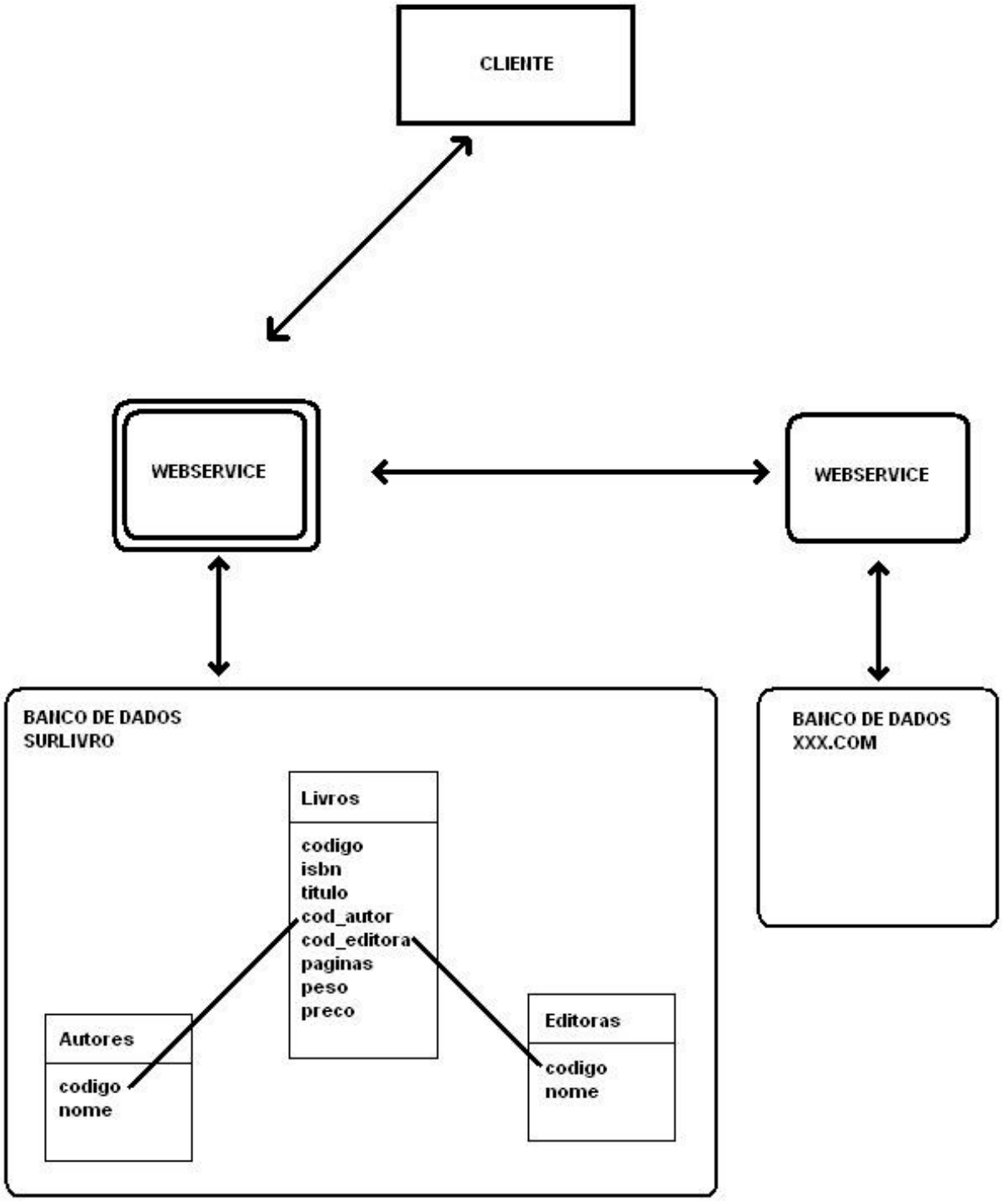

Figura 10: Modelo do fluxo de execução quando chamado o web service.

3.3 IMPLEMENTANDO O WEB SERVICE

Como a principal vantagem oferecida pela tecnologia Web Service é sua interoperabilidade com outras linguagens, a linguagem escolhida para o desenvolvimento desses serviços fica a critério do fabricante escolher com qual linguagem ele melhor se adapta e possui mais intimidade para desenvolver a solução para apresentar o resultado esperado.

Embora a linguagem escolhida para implementação não possua um suporte nativo a Web Services, PHP vem sendo muito utilizada pela sua simplicidade na criação e manutenção.

Algumas bibliotecas foram criadas em PHP para solucionar o suporte na criação de Web Services, tais como: PEAR::SOAP, NuSOAP e XML-RPC.

Para a implementação desse serviços foi utilizado a biblioteca NuSOAP, devido a sua ótima implementação e também por oferecer suporte na criação de WSDL, otimizando muito o tempo de desenvolvimento.

Para começar a implementação foi realizado o download da biblioteca NuSOAP que fará a comunicação SOAP entre nosso Web Service. Para realizar esse procedimento foi acessado o site http://sourceforge.net/projects/nusoap/ e realize o download da API. Logo em seguida já é possível começar a codificação do programa.

O Web Service implementado consta apenas com um único métodos. Sua execução é feita com a chamada do método getLivro. Esse, por sua vez, exige que seja passado dois valores como parâmetros, o VALOR, busca a ser encontrada, e FILTRO, pré-definidos como título, isbn, autor, e editora, especificando melhor a consulta.

A execução desse método implica em duas possibilidades de retorno. Caso sejam encontrados os livros referentes à busca solicitada, retorna-se uma lista de livros com suas respectivas informações. Em caso de não encontrar nenhum resultado, retorna-se uma lista vazia.

Na implementação do Web Service proposto neste trabalho, foi utilizado uma classe chamada db.php que faz toda a persistência no banco de dados da SUR O banco de dados utilizado neste caso é o mySQL.

O Web Service foi testado em dois servidores diferentes. Uma fonte ficou no servidor da Webnow, aonde é hospedado o website da SUR, e outra ficou no servidor da Hosted, aonde é hospedado o site www.mixingbeat.com. O acesso aos servidores ocorreu normalmente, não apresentando nenhum erro na execução da aplicação.

3.4 CRIANDO UM CLIENTE E CONSUMINDO O SERVIÇO

Para criar um cliente que acesse o Web Service criado é bem simples. Utilizando a classe NuSOAP você deve instanciar um objeto da mesma e realizar a chamada ao método disponível no serviço, como descreve o código abaixo:

```
<?php
//inclusão da classe NuSOAP
require_once('nusoap.php');

// Definição da localização do arquivo WSDL
$wsdl = 'http://www.mixingbeat.com/SOAP/server2.php?wsdl';

// criação de uma instância do cliente
$client = new soapclient($wsdl, true);

$result = $client->call('getLivro', array($busca, $filtro) );

// verifica se ocorreu falha na chamada do método
if ($client->fault){
        echo "Falha<_ré>".print_r($result)."</_ré>";
}
else{
// verifica se ocorreu erro
        $err = $client->getError();
        if ($err){
                echo "Erro<_ré>".$err."</_ré>";
        }
        else{
                //print_r($result);
        }//end_else
```

```
}//end_else
?>
```

A Figura 11 demonstra como é a tela que aparece ao cliente quando realiza uma pesquisa.

Figura 11: Tela do Cliente ao realizar uma pesquisa

4 CONSIDERAÇÕES FINAIS

Cada vez mais as empresas vêm buscando novas tecnologias que trabalhem de maneira integrada às já existentes, de forma simples, flexível e confiável. Como apresentado neste trabalho, isso se torna adequado quando se faz uso da tecnologia Web Service, que propõe a interoperabilidade dos sistemas de forma simples utilizando trocas de informações baseadas em XML.

Apesar do fato dessa tecnologia dispor de um forte amadurecimento no que se diz respeito à integração de sistemas, ela não é a solução para a comunicação entre todos os aplicativos.

O desenvolvimento do Web Service apresentado neste trabalho demonstrou a importância de sistemas diferentes se comunicarem utilizando informações que antes eram restritas apenas à empresa. É de grande ressalto, também, perceber que sistemas disponibilizados na Internet oferecem um ganho enorme à produtividade, pela possibilidade de ser acessado de qualquer lugar.

Sugere-se a elaboração de um Web Service mais robusto, que não realize apenas a cotação de livros, mas também a venda dos mesmos.

5 REFERÊNCIAS

FERREIRA, Cláudio. *Admirável Mundo Novo.*
Disponível em http://www.revistanetwork.com.br/shared/print_story.asp?id=23377.

W3C. Extensible Markup Language (XML).
Disponível em http://www.w3.org/XML.
Acessado em 03/02/2007.

Web Services Description Language
Disponível em http://www.w3.org/TR/wsdl
Acessado em 14/05/2007.

SOAP Specifications
Disponível em http://www.w3.org/TR/soap/
Acessado em 14/05/2007.

Assessing Your Legacy System.
Disponível em http://www.ibm.com/developerworks/rational/library/389.html
Acessado em 05/06/2007.

BACILI, Kleber R. Web Services e a integração de sistemas. ComputerWorld
Disponível em http://www.computerworld.com.br.
Acessado em 13/05/2007.

CLABBY, J. Web Services Explained - Solution and Applications for the real world. New York: Prentice Hall PTR. 2002.

SIQUEIRA, José Roberto B. Source Inspector – uma ferramenta para extração de regras de negócio em sistemas legados. Monografia (Mestrado em Informática). UFRJ /IM / NCE. Rio de Janeiro. 2002.

UDDI
Disponível em http://www.uddi.org/
Acessado em 07/06/2007.

SOA and Web Services
Disponível em http://java.sun.com/developer/technicalArticles/WebServices/soa/
Acessado em 11/06/2007.

6 APENDICE

O código descrito abaixo descreve a implementação realizada no lado do servidor para que possam ser realizadas as consultas quando acessado o Web Service pelo cliente.

```
<?php

// inclusão do arquivo de classes NuSOAP
require_once('nusoap.php');

// inclusão do arquivo que faz o acesso ao banco de dados local
include ("db.php");

// criação de uma instância do servidor
$server = new soap_server;

// inicializa o suporte a WSDL
$server->configureWSDL('tcc.livros','urn:tcc.livros');
$server->wsdl->schemaTargetNamespace = 'urn:tcc.livros';

//Cria um Tipo Complexo para a representação dos dados
$server->wsdl->addComplexType('Livros','complexType','struct','all','',
  array(
'isbn'=>array('name'=>'isbn','type'=>'xsd:string'),
'titulo'=>array('name'=>'titulo','type'=>'xsd:string'),
'autor'=>array('name'=>'autor','type'=>'xsd:string'),
'editora'=>array('name'=>'editora','type'=>'xsd:string'),
'preco'=>array('name'=>'preco','type'=>'xsd:string'),
'disponibilidade'=>array('name'=>'disponibilidade','type'=>'xsd:string')));

//Registra o tipo Lista de livros
```

```
$server->wsdl->addComplexType(“ListaLivros”, “complexType”, “array”, “”, “SOAP-ENC:Array”, array(),
    array(
       array(“ref”=>”SOAP-ENC:arrayType”,”wsdl:arrayType”=>”Livros[]”) ),
            “tns:Livros”);

    // Registra o nome método a ser oferecido
    $server->register(‘getLivro’, //nome do método
    array(‘valor’ => ‘xsd:string’, ‘filtro’ => ‘xsd:string’), //parâmetros de entrada
    array(‘livros’ => ‘tns:ListaLivros’), // parâmetros de saída
    ‘urn:server.livro’, //namespace
    ‘urn:server.livro#getLivro’, //soapaction
    ‘rpc’, //style
    ‘encoded’, //use
    ‘Passa-se o VALOR para pesquisa e retorna uma lista de livros com todas suas informações.’ //documentação do serviço
    );
    function getLivro($valor, $filtro) {
    $result;
    if($filtro == ‘autor’){
    $result = query(‘SELECT isbn, titulo, editora.nome as nomeEditora, autor.nome as nomeAutor, _ré_o, qtd FROM livros, editora, autor WHERE autor.nome like “%’.$valor.’%” AND livros.cod_editora=editora.codigo AND livros.cod_autor=autor.codigo ORDER BY titulo ‘);
    }
    else if($filtro == ‘editora’){
            $result = query(‘SELECT isbn, titulo, editora.nome as nomeEditora, autor.nome as nomeAutor, _ré_o, qtd FROM livros, editora, autor WHERE editora.nome like “%’.$valor.’%” AND livros.cod_editora=editora.codigo AND livros.cod_autor=autor.codigo ORDER BY titulo ‘);
    }
```

```
        else if($filtro == 'isbn' || $filtro == 'titulo'){
                $result = query('SELECT isbn, titulo, editora.nome as nomeEditora, autor.nome
as nomeAutor, _ré_o, qtd FROM livros, editora, autor WHERE livros.'.$filtro.' Like
"%'.$valor.'%" AND livros.cod_editora=editora.codigo AND livros.cod_autor=autor.codigo
ORDER BY titulo ');
        }

        if($result!=null){
                $cont = 0;
                $retorno = array();
                foreach($result as $r){
                        $isbn = $r->isbn;
                        $titulo = $r->titulo;
                        $autor = $r->nomeAutor;
                        $editora = $r->nomeEditora;
                        $preco = $r->_ré_o;
                        $retorno[$cont] = array('isbn'=>$isbn, 'titulo'=>$titulo, 'autor'=>$autor,
'editora'=>$editora, 'preco'=>$preco, 'disponibilidade'=>'Imediata');
                        $cont++;

                }
        }
        else{
                return facaCotacaoAmazon($valor);

        }
                return $retorno;
        }
```

// Método que acessa o Web Service da AMAZON.COM

```
function facaCotacaoAmazon($valor){

$client = new soapclient('http://soap.amazon.com/onca/soap3');
$namespace = 'http://soap.amazon.com';
$action = 'http://soap.amazon.com';

$params = array(
        'keyword' => $valor, // valor a ser pesquisado
        'page' => '1', // numero de paginas da consulta
        'mode' => 'books', // categoria a ser pesquisada
        'tag' => '0Y1V7W5V724JRSXXVMR2', // tag de identificação
        'type' => 'lite', // tipo da consulta: lite ou heavy
        'devtag' => '0Y1V7W5V724JRSXXVMR2', // identificação junto à Amazon
        'locale' => 'us' //moeda a ser utilizada
);

// nome do método de acesso ao Web Service da Amazon.com
$method = "KeywordSearchRequest";

// chamada do método do web service da Amazon.com
$result = $client->call($method,   array('KeywordSearchRequest' => $params),
        $namespace, $action);

        $cont=0;
foreach($result['Details'] as $detail){
        $isbn = $detail['Asin'];
        $titulo = $detail['ProductName'];
        $autor = $detail['Authors'][0];
        $editora = $detail['Manufacturer'];
        $preco = $detail['ListPrice'];
```

```
            $retorno[$cont] = array('isbn'=>$isbn, 'titulo'=>$titulo, 'autor'=>$autor,
'editora'=>$editora, 'preco'=>$preco, 'disponibilidade'=>'Aproximadamente 40 dias.');
            $cont++;
        }

        return $retorno;

        }

        // requisição para uso do serviço
        $HTTP_RAW_POST_DATA = isset($HTTP_RAW_POST_DATA) ?
$HTTP_RAW_POST_DATA : '';
        $server->service($HTTP_RAW_POST_DATA);
        ?>
```

A biblioteca NuSOAP fornece suporte automático para a criação da WSDL. Segue abaixo o modelo de WSDL gerado pela mesma, definindo o Web Service apresentado no trabalho.

```
        <?xml version="1.0" encoding="ISO-8859-1" ?>
         <definitions xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:SOAP-
ENC="http://schemas.xmlsoap.org/soap/encoding/" xmlns:tns="urn:tcc.livros"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns="http://schemas.xmlsoap.org/wsdl/"
targetNamespace="urn:tcc.livros">
         <types>
         <xsd:schema targetNamespace="urn:tcc.livros">
          <xsd:import namespace="http://schemas.xmlsoap.org/soap/encoding/" />
          <xsd:import namespace="http://schemas.xmlsoap.org/wsdl/" />
         <xsd:complexType name="Livros">
```

```
<xsd:all>
 <xsd:element name=”isbn” type=”xsd:string” />
 <xsd:element name=”titulo” type=”xsd:string” />
 <xsd:element name=”autor” type=”xsd:string” />
 <xsd:element name=”editora” type=”xsd:string” />
 <xsd:element name=”preco” type=”xsd:string” />
 <xsd:element name=”disponibilidade” type=”xsd:string” />
 </xsd:all>
 </xsd:complexType>
<xsd:complexType name=”ListaLivros”>
<xsd:complexContent>
<xsd:restriction base=”SOAP-ENC:Array”>
 <xsd:attribute ref=”SOAP-ENC:arrayType” wsdl:arrayType=”Livros[]” />
 </xsd:restriction>
 </xsd:complexContent>
 </xsd:complexType>
 </xsd:schema>
 </types>
<message name=”getLivroRequest”>
 <part name=”valor” type=”xsd:string” />
 <part name=”filtro” type=”xsd:string” />
 </message>
<message name=”getLivroResponse”>
 <part name=”livros” type=”tns:ListaLivros” />
 </message>
<portType name=”tcc.livrosPortType”>
<operation name=”getLivro”>
 <documentation>Passa-se o VALOR para pesquisa e retorna uma lista de livros com
todas suas informações.</documentation>
 <input message=”tns:getLivroRequest” />
 <output message=”tns:getLivroResponse” />
```

```
 </operation>
 </portType>
<binding name=”tcc.livrosBinding” type=”tns:tcc.livrosPortType”>
 <soap:binding style=”rpc” transport=”http://schemas.xmlsoap.org/soap/http” />
<operation name=”getLivro”>
 <soap:operation soapAction=”urn:server.livro#getLivro” style=”rpc” />
<input>
 <soap:body use=”encoded” namespace=”urn:server.livro”
encodingStyle=”http://schemas.xmlsoap.org/soap/encoding/” />
 </input>
<output>
 <soap:body use=”encoded” namespace=”urn:server.livro”
encodingStyle=”http://schemas.xmlsoap.org/soap/encoding/” />
 </output>
 </operation>
 </binding>
<service name=”tcc.livros”>
<port name=”tcc.livrosPort” binding=”tns:tcc.livrosBinding”>
 <soap:address location=”http://www.mixingbeat.com/SOAP/server2.php” />
 </port>
 </service>
 </definitions>
```