Revista EDUCAÇÃO E TECNOLOGIA
Ano 1 - Número 2 - Out/Mar - 2006

CONSTRUÇÃO DE APLICAÇÕES DISTRIBUÍDAS UTILIZANDO SERVIÇOS WEB

Deusa Cesconeti e Jean Eduardo Glazar

Departamento de Ciência da Computação
Faculdade de Aracruz – UNIARACRUZ
{dcescone, jean}@fsjb.edu.br

RESUMO

Apresenta um conjunto de tecnologias, chamado serviços web (*Web Services*), para suporte às Aplicações Orientadas a Serviços. Essas tecnologias permitem o desenvolvimento de aplicações distribuídas que possam reutilizar códigos, diminuindo os custos de desenvolvimento. Elas podem ser utilizadas entre ambientes heterogêneos (multiplataformas), facilitando a utilização em sistemas já implementados. O responsável por essa independência é o protocolo SOAP, mas os serviços web utilizam também XML, WSDL e UDDI, que serão todos descritos. Também apresenta algumas ferramentas úteis para o processo de publicação e acesso aos serviços.

Palavras-chave: Serviços web. Aplicações orientadas a serviços.

ABSTRACT

In this paper we present a range of technologies, called Web Services, to support Service Oriented Architectures (SOA). Web services are considered a promising technology in the process of integrating distributed applications. These technologies allow the development of distributed applications that can reuse codes to minimize the development costs. They can be used between heterogeneous environments, facilitating the use in already implemented systems. The SOAP protocol is the main responsible for this independence features, but the Web Services also use XML, WSDL and UDDI. This paper will describe all those technologies and some useful tools to publish and access the services.

Keywords: Web services. Services oriented architecture.

INTRODUÇÃO

Este trabalho tem como objetivo apresentar um conjunto de tecnologias, chamadas serviços web (*Web Services*), para suporte a aplicações orientadas a serviços. Essa tecnologia está sendo muito utilizada devido às suas vantagens, por exemplo, a sua adequação a ambientes heterogêneos (multiplataformas) como a Internet. Outra vantagem é que permite que os códigos possam ser reutilizados (veja em Web Services Architect).

Para desenvolver alguma aplicação utilizando serviços web, é necessário conhecer outras tecnologias que compõem a arquitetura em quatro camadas: *Extensible Markup Language* (XML), *Simple Object Application Protocol* (SOAP), *Web Service Definition Language* (WSDL) e *Universal Discovery Description Integration* (UDDI), que são responsáveis pela publicação e localização dos serviços (veja em W3C - Web Services Architecture).

SERVIÇOS WEB

Os serviços web são aplicações de software que podem ser acessadas remotamente, usando diferentes linguagens baseadas em XML. Podem ser publicados, localizados e chamados pela Internet. Normalmente, os serviços web são identificados por um *Uniform Resource Locators* (URL), exatamente como qualquer outro site web. O que torna os serviços web diferentes dos sites web comuns é a capacidade de integração de várias plataformas.

A utilização de serviços web é atrativa, em parte, pela sua adequação a ambiente de computação heterogêneo e a facilidade que eles têm para o encapsulamento de funções de negócio (COSTA, 2004), e se caracterizam por serem fracamente acoplados e altamente interoperáveis (SILVA, 2004).

A comunicação entre cliente e servidor pode ocorrer utilizando sistemas operacionais diferentes, entre ambientes heterogêneos. O responsável por essa independência é o protocolo SOAP.

Os servidores descrevem seus serviços no documento WSDL. Esse documento permite especificar a interface de serviços web, seus métodos, detalhes de transporte e do ponto de acesso aos serviços (COSTA, 2004). A partir dessa descrição, pode-se construir uma interface para o cliente em sua linguagem nativa,

que atenda às especificações dos serviços descritos. Com isso os clientes facilmente obtêm informações sobre os servidores que usarão (estrutura, método e parâmetros exigidos). Isso é útil para a codificação de aplicações servidoras que serão utilizadas por terceiros ou para implementação de aplicações a clientes que usam serviços de outras empresas.

Existem algumas organizações que controlam as especificações dos serviços web, porém a mais importante dessas organizações é o W3C (*World Wide Web Consortium* - http://www.w3.org), o qual desenvolve tecnologias interoperáveis (especificações, instruções, *software* e ferramentas), para levar a *web* ao seu potencial máximo. O W3C é um fórum de informações, comércio, comunicação e aprendizado coletivo, e é ele quem controla as especificações SOAP, WSDL, XML, entre outras.

VANTAGENS DE UTILIZAR SERVIÇOS WEB

Potts (2003) descreve muitas vantagens para utilizar a arquitetura dos serviços web:

- Integração entre plataformas heterogêneas. Aplicações em sistemas operacionais diferentes podem comunicar-se sem problemas, devido à especificação do protocolo SOAP.
- A interconexão entre sistemas legados (estabelecidos, seguros e operacionais) é bem feita nos serviços web, independente da linguagem em que eles são escritos. Os serviços web podem ser escritos de uma maneira que exija pouca ou nenhuma mudança na base de código legada. O resultado final é um moderno sistema distribuído que conserva toda a base de código legada que a empresa construiu.
- O custo com desenvolvimento de *software* é menor, pois, com os serviços web, não há a necessidade de reinventar código, este é feito uma vez e utilizado quantas vezes for preciso.
- O alto nível de compromisso dos fornecedores, tanto de *hardware* quanto de *software*, com essa tecnologia, diminuiu o custo do seu uso. A concorrência não só baixa os preços das ferramentas, mas também encoraja a inovação.

- Com os serviços web, é possível a integração com seus parceiros comerciais. Ao interligar o sistema de uma empresa com os seus fornecedores, por exemplo, facilitará a aquisição de mercadorias a qualquer hora.
- As funções do negócio ficam encapsuladas no servidor, o que permite que os clientes fiquem livres das regras de negócio. Qualquer manutenção que seja necessária será realizada apenas no servidor, mantendo a mesma interface de comunicação com o cliente.

DESVANTAGENS DE UTILIZAR SERVIÇOS WEB

Embora os serviços web apresentem diversas vantagens para a sua utilização, eles ainda não são a solução mágica para todos os problemas (POTTS, 2003).

- O primeiro problema é sobre a segurança. As informações enviadas pela Internet podem ser vistas por outras pessoas. Existe um programa de *Secure Socket Layer* (SSL) para enviar informações confidenciais para serviços web, mas ele também não deixa de ter problemas. Ele é lento, pois criptografa todo o pacote SOAP, não sendo indicado para grande transferência de dados. O W3C já possui projetos para resolver esse problema.
- A transação sem controle também é outro problema. Se o sistema precisar confirmar que a transação foi feita, ou se houver algum problema no caminho e for preciso que o processo seja desfeito, isso terá que ser implementado no próprio programa, pois os serviços web não possuem esse controle.
- O meio de transmissão no serviços web não é confiável, o que faz com que uma transação fique perdida no meio do caminho, sem que haja um retorno, se houver algum problema. Isso ocorre porque o protocolo HTTP não é um protocolo seguro, uma vez que não garante entrega ou resposta.

Existem maneiras de corrigir esses problemas, mas essa responsabilidade cabe ao desenvolvedor do sistema. Segundo Silva (2004), esse é um grande problema, porque cria preocupações que não são diretamente relacionadas com as regras de negócio das aplicações, gerando um aumento desnecessário no tamanho da complexidade dos programas. No entanto, o W3C trabalha para aperfeiçoar as tecnologias dos serviços web.

ARQUITETURA DOS SERVIÇOS WEB

A *Service-Oriented Architecture* (SOA) fornece o modelo teórico para todos os serviços web (COSTA, 2004). Trata-se de um modelo simples, que contém três entidades e três operações, conforme Figura 1 (W3C - Web Services Architecture).

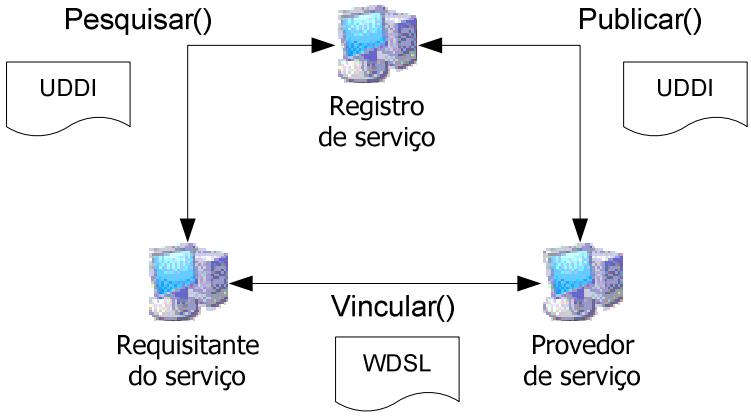

Figura 1 - Três componentes básicos utilizados pelos serviços web

Observando a Figura 1, tem-se:

1. O **provedor de serviços** é onde estão os serviços web. O primeiro passo é **publicar** as informações desses serviços, detalhes de classificação, detalhes de conexão e outras informações, no registro dos serviços. Essas informações são especificadas pela UDDI. Nesse passo, o provedor de serviços também gera o documento WSDL com os detalhes dos serviços, formas de conexão, métodos, parâmetros e tipo de retorno. O documento WSDL será usado pelo requisitante de serviços para se **vincular** ao provedor de serviços.
2. O **registro de serviços** é uma aplicação que retorna informações sobre os serviços web em resposta a uma requisição que tenha sido submetida por um requisitante de serviços na operação **Pesquisar()**. Essas informações (UDDI) retornam os detalhes de contato para os serviços web, com base em suas classificações que correspondem ao critério de pesquisa. Esse registro também inclui informações sobre como descobrir os detalhes de conexão.
3. O **requisitante de serviços** é uma aplicação que deseja utilizar os serviços publicados pelo provedor. Ele não sabe onde estão os serviços, nem como encontrá-los; então, recorre ao registro de serviços e requisita a operação

pesquisar. De posse da UDDI, consegue descobrir onde estão os serviços, então se conecta ao provedor e obtém o documento WSDL, com os detalhes dos serviços. Com a aplicação pronta pode se **vincular** aos serviços no provedor.

Segundo Costa (2004), um dos objetivos do uso da SOA é prover o desenvolvimento de aplicações distribuídas pela composição rápida e de baixo custo de componentes de *software* autônomos e independentes de plataforma.

Os serviços web, para fazerem a comunicação entre as aplicações, utilizam quatro camadas que empacotam a requisição e a resposta entre cliente e servidor. As camadas utilizadas são (W3C - Web Services Glossary):

- *Extensible Markup Language* (XML);
- *Simple Object Application Protocol* (SOAP);
- *Web Service Definition Language* (WSDL);
- *Universal Discovery Description Integration* (UDDI).

UNIVERSAL DISCOVERY DESCRIPTION INTEGRATION (UDDI)

A camada UDDI é um padrão desenvolvido para os serviços web publicarem informações e anunciarem os seus serviços de forma rápida e fácil. Ela contém informações no sentido de permitir a pesquisa de certo conjunto de características ou pelo conjunto de recursos que se deseja utilizar.

Essa especificação descreve um registro que lista serviços web pelos quais um cliente possa estar interessado. Esses registros podem ser de vários tipos:

- **públicos:** quando o registro é público, ele está aberto para qualquer pessoa pesquisar, não possui nenhuma restrição;
- **privados:** quando o registro é privado, ele está por trás de um *firewall* de uma organização. Não é qualquer pessoa que pode fazer pesquisa, é voltado para pesquisas internas de uma determinada organização;
- **restritos:** quando o registro é restrito, somente organizações que têm permissão para acessá-lo poderão fazer pesquisas.

WEB SERVICE DESCRIPTION LANGUAGE (WSDL)

A WSDL (W3C – WSDL) é uma interface que define os métodos pertencentes aos serviços. Não é necessário na programação, serve apenas para especificar o "contrato" que o cliente deverá cumprir para utilizar os serviços. Um documento WSDL é um documento XML.

Ele é independente de plataforma e linguagem. Esse documento pode ser lido por um programa ou por um programador, os quais podem criar mensagens claras que podem chamar um ou mais métodos nesses serviços. Existem diversas ferramentas para gerar e ler um documento WSDL.

O servidor gera o documento WSDL baseado nos seus serviços e disponibiliza-o em um endereço na Internet. O cliente acessa o documento WSDL, para saber quais são os métodos disponíveis e como acessá-los, e gera a interface desses métodos na linguagem que será utilizada para criar a aplicação cliente. A partir de então, a aplicação cliente usa esses métodos como se eles estivessem em seu computador local.

SIMPLE OBJECT ACCESS PROTOCOL (SOAP)

A comunicação entre um cliente e um servidor é feita por meio do SOAP, que é definido em XML. É um protocolo elaborado para facilitar a chamada remota de funções via Internet, permitindo que dois programas se comuniquem de uma maneira semelhante à invocação de páginas Web. Esse protocolo está se tornando padrão para a troca de mensagens entre aplicações e serviços web (W3C – SOAP).

Para o transporte das mensagens SOAP, é usado o protocolo HTTP, que torna o SOAP um protocolo leve.

Os pedidos SOAP podem ser feitos em três padrões: GET, POST e SOAP. Os padrões GET e POST são iguais aos pedidos feitos por navegadores da Internet. Já o SOAP é semelhante ao POST, mas os pedidos são feitos em XML.

Por ser bastante simples, o SOAP é definido em um envelope que contém: cabeçalho (*header*) e corpo (*body*). No cabeçalho constam todas as informações necessárias do aplicativo. O corpo contém o *payload*, que é a chamada de métodos do originador e é a resposta do sistema remoto.

O programador não precisa criar os pacotes SOAP, pois existem ferramentas responsáveis pelo envio e recebimento de pacotes SOAP. A aplicação cliente apenas chama os métodos como se eles estivessem no mesmo computador. Sempre que uma chamada a um serviço web é executada, o pacote SOAP é criado e enviado para o servidor, que chamará o método correspondente e enviará outra mensagem de retorno.

EXTENSIBLE MARKUP LANGUAGE (XML)

A base da comunicação nos serviços web é o XML (W3C – XML). É uma linguagem similar ao HTML, mas com finalidade diferente. O HTML descreve como um documento deverá ser exibido em um navegador. Já o XML descreve o significado dos dados, independente de como eles serão exibidos. Essa se tornou uma linguagem comum na computação.

O XML é um formato de texto muito simples e flexível. Originalmente foi projetado para encontrar chaves em documentos eletrônicos em larga escala. O XML está também se tornando um importante padrão para a troca de uma vasta variedade de dados na web. Ele é uma linguagem que pode ser entendida tanto por um ser humano (programador), quanto por outros programas. Por isso ninguém precisa ser experiente em XML. Existem pacotes de programas para gerar e ler arquivos em XML. Por meio do XML, é possível estabelecer objetos, métodos, dados e seus respectivos tipos que serão interpretados pela aplicação destino.

APLICAÇÃO SERVIDORA

A aplicação servidora será aquela que publicará todos os serviços web de um determinado negócio. Os serviços web, para serem publicados, terão que ser executado em uma máquina contendo uma infra-estrutura necessária.

Não basta apenas implementar os serviços web, é necessário publicá-los e inicializá-los para que eles possam ser utilizados. A classe responsável por **publicar** os serviços web cria o endereço que possui os serviços. Nesse momento, é gerado o documento WSDL, que possui a forma-padrão para a comunicação dos dados.

Existem diversas ferramentas para gerar o documento WSDL e publicar os serviços. Neste trabalho, será descrito o pacote **Systinet,** que disponibiliza um conjunto de classes que permite gerar e acessar o documento WSDL.

A classe que irá registrar e iniciar os serviços web será a ***WebServiceStartup***, pertencente ao pacote Systinet, a qual apresenta mecanismo de publicação dos métodos utilizando o *Web Applications and Services Plataform* (WASP), que é uma plataforma que oferece mecanismos de desenvolvimento tanto para C++ quanto para Java. Nesse exemplo, foi utilizada a linguagem Java. A classe *WebServiceStartup* deve ser alterada para cada conjunto de serviços web a ser publicado, conforme código no Quadro 1.

```
private static String serverURL = "http://10.0.0.107:6060/";
private static String serviceEndpoint = "/Matriz/";
private static Class  toPublish = Matriz_Web.class;

// Primeiro cria e inicializa o serviço no Wasp Server...
   Transport http = Wasp.startServer(serverURL);

// Agora o serviço é publicado...
   Registry.publish(serviceEndpoint, toPublish);
```

Quadro 1 - Comandos para publicar os serviços web

O objeto **serverURL** é o endereço do computador onde estarão os serviços web a serem publicados. O objeto **serviceEndpoint** é o nome dado para identificar os serviços web. O objeto **toPublish** é o nome da classe onde estão implementados os serviços web.

Para que o código do Quadro 1 seja executado, será necessário incluir as seguintes bibliotecas do WASP Server:

- org.systinet.wasp.webservice.Registry;
- org.systinet.wasp.Wasp;
- org.idoox.transport.Transport;
- org.idoox.transport.TransportStartException.

O documento WSDL, contendo as informações sobre os métodos, é gerado no momento da publicação dos serviços web (class *WebServiceStartup*).

Os serviços estando publicados e em execução, a aplicação cliente poderá fazer a chamada a qualquer método que esteja publicado no servidor, pelo endereço especificado na URL.

APLICAÇÃO CLIENTE

Para que a aplicação cliente possa fazer a chamada remota dos métodos no servidor, será necessário obter a descrição dos serviços especificados no documento WSDL e criar a interface, com base nessas descrições, para a linguagem que será usada para implementar a aplicação cliente, nesse caso, em interface Java. Para que isso seja realizado, a plataforma **Systinet** disponibiliza, pelo pacote **wasp.jar**, um conjunto de classes que permite o acesso a métodos remotos e a ferramenta WSDL2Java, que obtém a descrição WSDL das classes publicadas nos serviços web, referenciando as transformações em interface Java. Para que tudo funcione, a máquina cliente precisa ter o **Systinet** instalado com a versão igual ou superior a que o servidor esteja utilizando.

Para obter o documento WSDL com as descrições dos serviços e criar a interface na aplicação cliente, será necessário executar o comando pertencente ao pacote do Systinet:

```
wsdl2java –u http://10.0.0.107:6060/Matriz
```

Onde: “http://10.0.0.107:6060/Matriz” é a URL do servidor, onde o documento WSDL foi gerado com as informações para acesso aos serviços web.

Após o mapeamento do documento WSDL em interface Java, a aplicação cliente pode ser implementada, utilizando os métodos do servidor como se fossem métodos locais.

TESTES

Para a realização de testes utilizando os serviços web, foi escolhida a plataforma Windows e o pacote de ferramentas para os serviços web **Systinet 5.5.** O pacote Systinet deve ser instalado tanto no servidor quanto no cliente, em versões compatíveis, no caso, a mesma versão. Ao final da instalação do **Systinet,** é necessário alterar o arquivo **env.bat** que se encontra no diretório de instalação do systinet, **c:\Systinet\server_java55\bin**, conforme o Quadro 2:

```
@ECHO OFF
IF "%JAVA_HOME%" == "" (
  ECHO JAVA_HOME must be set!
  PAUSE
  GOTO :EOF
)
SET WASP_HOME=c:\systinet\server_java55
REM WASP_HOME
SET JAVA_CMD=%JAVA_HOME%\java
SET BUILD_FILE=build-default.xml
:end
```

Quadro 2 – Arquivo env.bat para uso em Windows

Para utilizar o Java, é necessário acrescentar as seguintes variáveis de ambiente no Windows, com o caminho de instalação do jdk, como exemplo:

set java_home = C:\Arquivos de Programas\JBuilder9\jdk1.4\bin

set Java_cmd = C:\Arquivos de Programas\JBuilder9\jdk1.4\bin

É preciso adicionar ao *path* da máquina os caminhos de instalação do Java e do Systinet, como exemplo: **C:\Arquivos de Programas\JBuilder9\jdk1.4\bin**; e **C:\systinet\server_java55\bin**.

CONCLUSÃO

Podemos observar que os serviços web são de grande utilidade, permitindo o desenvolvimento de sistemas distribuídos com menor custo e menor tempo, pois podemos utilizar códigos disponíveis e testados.

A integração entre plataformas heterogêneas é o grande diferencial dos serviços web, comparados com outras tecnologias para sistemas distribuídos. O responsável por essa integração é o protocolo SOAP. Por ser baseado em XML, ele é entendido em diversas plataformas.

Apesar de existirem sistemas que se comuniquem *on-line*, os serviços Web possuem as suas vantagens, por exemplo, a aplicação cliente fica livre das regras de negócio da aplicação servidora. Qualquer alteração que tenha que ser feita será realizada na aplicação servidora. Essa alteração é transparente para o cliente, desde que a interface entre os dois continue a mesma.

Porém, os serviços Web ainda estão em fase de aprimoramento e possuem algumas desvantagens, como não garantir a segurança dos dados e nem controle de transações, que hoje são fundamentais para qualquer sistema distribuído. Essas preocupações não são diretamente relacionadas com as regras de negócio das aplicações, gerando um aumento desnecessário no tamanho da complexidade dos programas. No entanto, o W3C trabalha para aperfeiçoar as tecnologias dos serviços web.

REFERÊNCIAS

1 COSTA, M. B.; SANTOS N., P.; RESENDE, R. P. **Utilização de aspectos no desenvolvimento de aplicações baseadas em Serviços Web.** WASP'2004 – I Workshop Brasileiro de Desenvolvimento Orientado a Aspectos. Brasília, outubro, 2004. Disponível em: <http://www.dcc.ufmg.br/~mcosta/>. Acesso em: 15 fev. 2005.

2 POTTS, Stephen; KOPAK, Mike. **Aprenda em 24 horas Web Services**. Rio de Janeiro, RJ: Campus, 2003.

3 SILVA, C.; MENDONÇA, N. C. **Uma abordagem para integrar aspectos e Serviços Web.** WASP'2004 – I Workshop Brasileiro de Desenvolvimento Orientado a Aspectos. Brasília, outubro, 2004. Disponível em: <http://twiki.im.ufba.br/pub/WAsp/AcceptedPapers/artigo_WASP_AspectosServicosWeb.pdf>. Acesso em: 15 fev. 2005.

4 SYSTINET. **Wasp Server for Java 5.0 product documention**: Systinet software. Disponível em: <http://www.systinet.com/doc/wasp_jserver/waspj/index.html>. Acesso em: 20 fev. 2005.

5 WEB SERVICES ARCHITECT. **Web Services Architect**. Disponível em: <http://www.webservicesarchitect.com/downloads.asp>. Acesso em: 20 fev.2005.

6 W3C. **Web Services Architecture** - W3C Working Draft. Disponível em: <http://www.w3.org/TR/2002/WD-ws-arch-20021114/>. Acesso em: 25 jan. 2005.

7 W3C. **Web Services Glossary** - W3C Working Draft. Disponível em: <http://www.w3.org/TR/2003/WD-ws-gloss-20030514/>. Acesso em: 25 jan. 2005.

8 W3C. **WSDL** – W3C Recommendation 3 August 2004. Disponível em: <http://www.w3.org/TR/wsdl>. Acesso em: 25 jan. 2005.

9 W3C. **SOAP** – W3C Recommendation 25 January 2005. Disponível em: <http://www.w3.org/TR/2005/REC-soap12-mtom-20050125>. Acesso em: 25 jan. 2005.

10 W3C. **XML** – W3C Recommendation 27 June 2001. Disponível em: <http://www.w3.org/TR/2001/REC-xmlbase-20010627>. Acesso em: 25 jan. 2005.