**Programando Serviços Web com SOAP**

**Capítulo 6. Descobrindo Serviços SOAP**

Uma vez que uma descrição WSDL de um serviço web tenha sido criada, um consumidor de serviço deve ser capaz de localizá-la para poder usá-la. Isso é conhecido como descoberta, o tema deste capítulo. Em particular, veremos o projeto Universal Description, Discovery, and Integration (UDDI) e a nova Linguagem de Inspeção de Serviços Web.

O WSDL fornece a um consumidor de serviço todas as informações necessárias para interagir com um provedor de serviço. Mas como um consumidor pode conhecer serviços para usar? O projeto UDDI é um esforço da indústria para definir um registro pesquisável de serviços e suas descrições, para que consumidores possam descobrir automaticamente os serviços de que precisam.

O UDDI tem duas partes: um registro de todos os metadados de um serviço web (incluindo um ponteiro para a descrição WSDL de um serviço) e um conjunto de definições de tipo de porta WSDL para manipular e pesquisar esse registro.

A especificação mais recente do UDDI é a Versão 2.0. Neste livro, no entanto, focamos completamente na Versão 1.0. A Versão 2.0 ainda não foi amplamente implementada e há muito pouco suporte disponível para ela.

O UDDI não é a única opção para descoberta de serviços. A IBM e a Microsoft anunciaram recentemente a Web Services Inspection Language (WS-Inspection), uma linguagem baseada em XML que fornece um índice de todos os serviços web em uma determinada localização da web.

A primeira parte deste capítulo se concentrará principalmente no UDDI. A última metade apresentará brevemente o WS-Inspection e demonstrará seu papel na Descoberta de Serviços.

**6.1 O Registro UDDI**

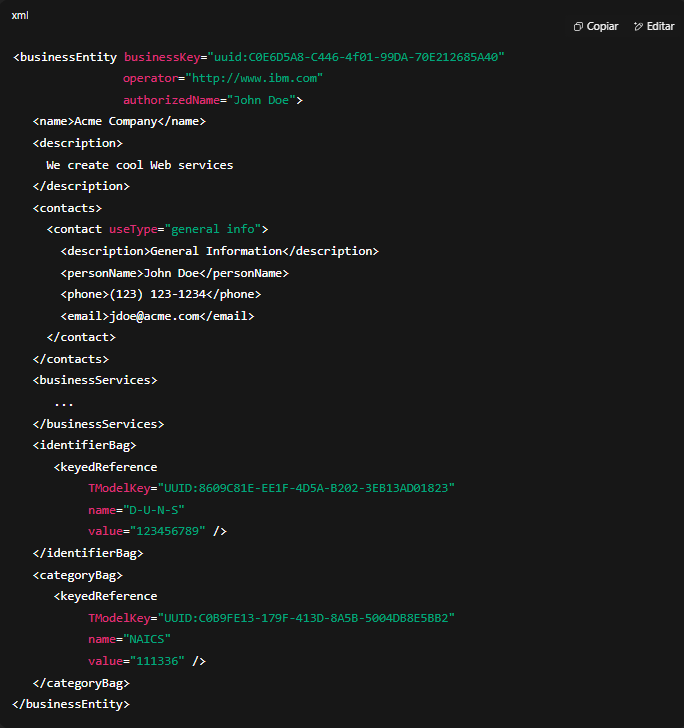
O registro UDDI permite que uma empresa liste publicamente uma descrição de si mesma e dos serviços que oferece. Consumidores potenciais desses serviços podem localizá-los com base em informações taxonômicas, como o que o serviço faz ou qual indústria ele atende.

O próprio registro é definido como uma hierarquia de descrições de negócios, serviços e vinculações expressas em XML.

**6.1.1 Entidade de Negócio**

A estrutura de entidade de negócio representa o provedor de serviços web. Dentro do registro UDDI, essa estrutura contém informações sobre a própria empresa, incluindo informações de contato, categorias da indústria, identificadores comerciais e uma lista de serviços oferecidos. O Exemplo 6-1 mostra a entrada fictícia de uma empresa no registro UDDI.

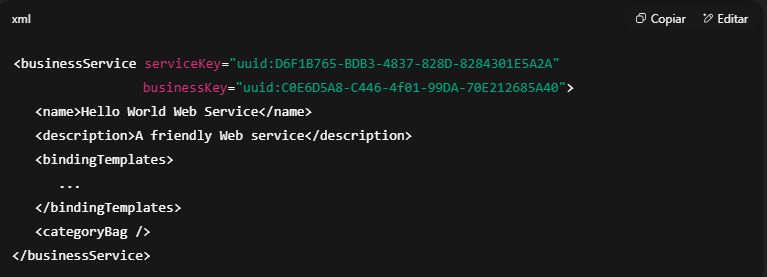
**Exemplo 6-1. Uma entrada de negócio no UDDI**



**6.1.2 Serviços de Negócio**

A estrutura de serviço de negócio representa um serviço web individual fornecido pela entidade de negócio. Sua descrição inclui informações sobre como se conectar ao serviço web, que tipo de serviço web é e a quais categorias taxonômicas ele pertence. O Exemplo 6-2 mostra uma possível estrutura de serviço de negócio para o serviço web Hello World.

**Exemplo 6-2. Estrutura de negócio Hello World no UDDI**

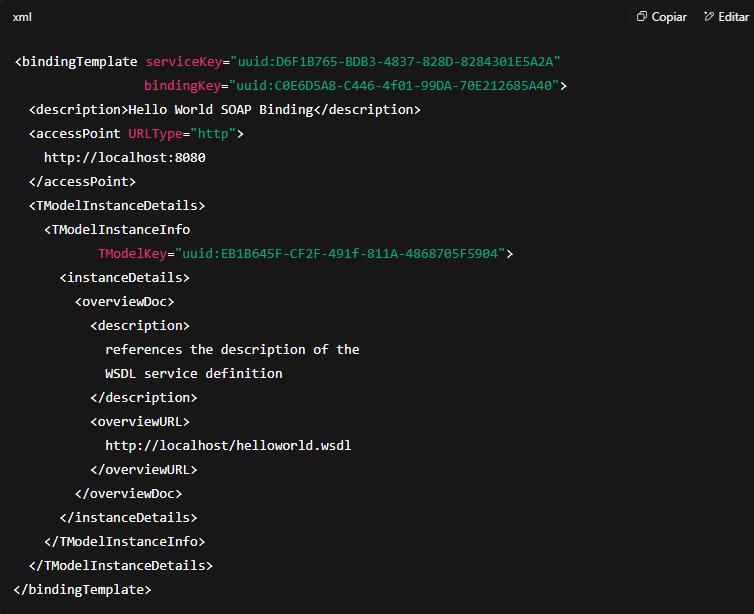


Observe o uso dos Identificadores Universalmente Únicos (UUIDs) nos atributos **businessKey** e **serviceKey**. Cada entidade de negócio e cada serviço de negócio é identificado de forma única em todos os registros UDDI através do UUID atribuído pelo registro quando a informação é inserida pela primeira vez.

**6.1.3 Modelos de Ligação (Binding Templates)**

Os modelos de ligação são as descrições técnicas dos serviços web representados pela estrutura de serviço de negócio. Um único serviço de negócio pode ter múltiplos modelos de ligação. O modelo de ligação representa a implementação real do serviço web (é aproximadamente equivalente ao elemento **service** que vimos no WSDL). O Exemplo 6-3 mostra um modelo de ligação para Hello World.

**Exemplo 6-3. Um modelo de ligação para Hello World**



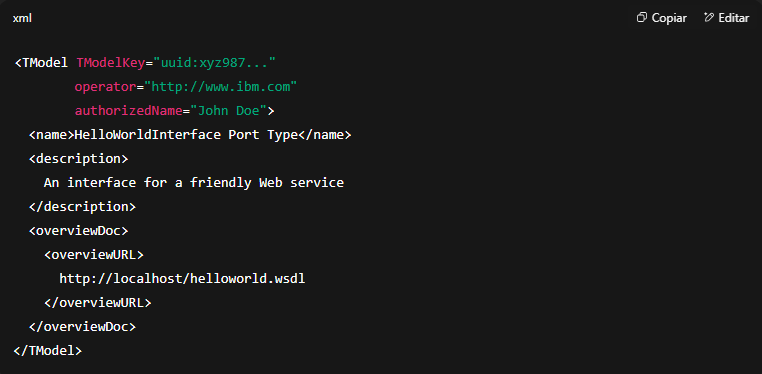
Como um serviço de negócio pode ter múltiplos modelos de ligação, o serviço pode especificar diferentes implementações do mesmo serviço, cada uma ligada a um conjunto diferente de protocolos ou a um endereço de rede diferente.

**6.1.4 TModels**

Um **TModel** é uma forma de descrever as várias estruturas de negócio, serviço e modelo armazenadas dentro do registro UDDI. Qualquer conceito abstrato pode ser registrado dentro do UDDI como um TModel. Por exemplo, se você definir um novo tipo de porta WSDL, poderá definir um TModel que represente esse tipo de porta dentro do UDDI. Então, você pode especificar que um determinado serviço de negócio implementa esse tipo de porta associando o TModel a um dos modelos de ligação desse serviço de negócio.

Um TModel representando o tipo de porta **HelloWorldInterface** é mostrado no Exemplo 6-4.

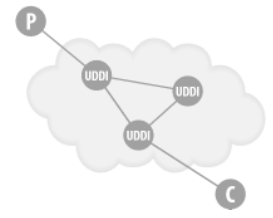
**Exemplo 6-4. Um TModel para Hello World**



**6.1.5 Registros UDDI Federados**

No seu núcleo, o UDDI é composto por uma rede global de registros interligados (federados) que implementam a mesma interface de serviço web baseada em SOAP para publicar e localizar serviços web. A Figura 6-1 ilustra isso.

**Figura 6-1.** Registros UDDI podem ser interligados para fornecer uma capacidade rudimentar de pesquisa distribuída



**6.1.6 Registros UDDI Privados**

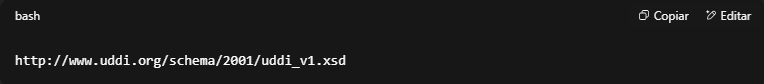
Como alternativa ao uso da rede pública federada de registros UDDI disponíveis na Internet, empresas ou grupos da indústria podem optar por implementar seus próprios registros UDDI privados. Esses serviços exclusivos seriam projetados com o único propósito de permitir que membros da empresa ou do grupo da indústria compartilhem e anunciem serviços entre si.

A chave para isso, entretanto, é que, seja o registro UDDI parte da rede global federada ou um registro de propriedade e operação privada, o que une tudo é uma API comum de serviços web para publicar e localizar empresas e serviços anunciados dentro do registro UDDI.

**6.2 As Interfaces do UDDI**

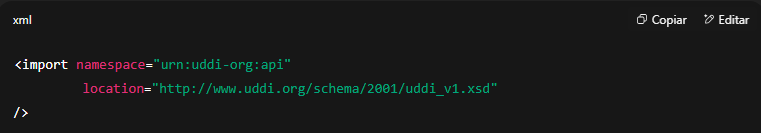
Um registro não serve de nada sem alguma maneira de acessá-lo. O padrão UDDI especifica duas interfaces SOAP para consumidores de serviços e provedores de serviços interagirem com o registro. Consumidores de serviços usam **InquireSOAP** para encontrar um serviço, e provedores de serviços usam **PublishSOAP** para listar um serviço. Esses serviços são descritos com WSDL. A explicação a seguir das APIs SOAP refere-se ao WSDL, mas abrevia algumas das partes repetitivas. A especificação completa do WSDL da API UDDI é fornecida no Apêndice B.

O núcleo das interfaces UDDI são as definições de Esquema XML do UDDI. Elas definem os tipos de dados fundamentais do UDDI, por exemplo, o **businessDetail**, que comunica informações detalhadas sobre entidades de negócio registradas. O Esquema XML do UDDI deve ser importado para a descrição WSDL a partir de sua localização na rede em:



como mostrado no Exemplo 6-5.

**Exemplo 6-5. Importando a descrição WSDL**



**6.2.1 A Interface Publisher**

A interface Publisher define dezesseis operações para um provedor de serviços gerenciar suas entradas no registro UDDI:

* **get\_authToken**  
  Recupera um token de autorização. Funciona exatamente como o token de autorização usado no exemplo do Publisher no Capítulo 3. Todas as operações da interface Publisher exigem que um token de autorização válido seja enviado com a solicitação.
* **discard\_authToken**  
  Instrui o registro UDDI a não aceitar mais um determinado token de autorização. Este passo é equivalente a encerrar a sessão no sistema.
* **save\_business**  
  Cria ou atualiza as informações de uma entidade de negócio contidas no registro UDDI.
* **save\_service**  
  Cria ou atualiza informações sobre os serviços web que uma entidade de negócio fornece.
* **save\_binding**  
  Cria ou atualiza as informações técnicas sobre a implementação de um serviço web.
* **save\_TModel**  
  Cria ou atualiza o registro de conceitos abstratos gerenciados pelo registro UDDI.
* **delete\_business**  
  Remove completamente as entidades de negócio especificadas do registro UDDI.
* **delete\_service**  
  Remove completamente os serviços web especificados do registro UDDI.
* **delete\_binding**  
  Remove os detalhes técnicos especificados de um serviço web do registro UDDI.
* **delete\_TModel**  
  Remove os TModels especificados do registro UDDI.
* **get\_registeredInfo**  
  Retorna um resumo de tudo o que o registro UDDI está atualmente acompanhando para o usuário, incluindo todos os negócios, todos os serviços e todos os TModels.

No WSDL, esses métodos correspondem a mensagens baseadas nos tipos de dados subjacentes do UDDI, como no Exemplo 6-6.

**Exemplo 6-6. Definição de método UDDI**



As outras mensagens padrão são definidas de maneira semelhante.

Finalmente, definimos o **port type** propriamente dito, criando a interface pela qual modificações podem ser feitas no registro UDDI. Novamente, apenas algumas definições são mostradas em detalhe no Exemplo 6-7, pois todas seguem o mesmo padrão.

**Exemplo 6-7. Definições representativas de operações Publisher**



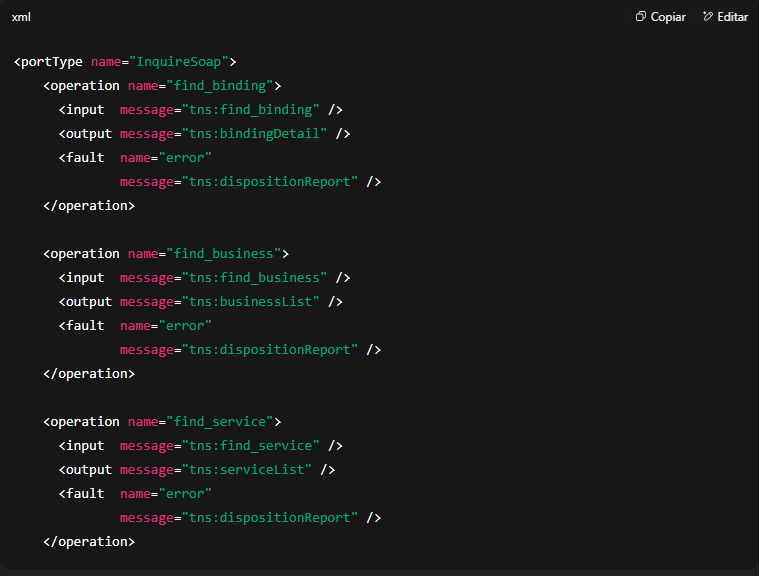
**6.2.2 A Interface Inquiry**

A interface Inquiry define dez operações para pesquisar o registro UDDI e recuperar detalhes sobre registros específicos:

* **find\_binding**  
  Retorna uma lista de serviços web que correspondem a um conjunto específico de critérios, com base nas informações técnicas de ligação.
* **find\_business**  
  Retorna uma lista de entidades de negócio que correspondem a um conjunto específico de critérios.
* **find\_service**  
  Retorna uma lista de serviços web que correspondem a um conjunto específico de critérios.
* **find\_TModel**  
  Retorna uma lista de TModels que correspondem a um conjunto específico de critérios.
* **get\_bindingDetail**  
  Retorna as informações completas de registro para um modelo de ligação (binding template) de serviço web específico.
* **get\_businessDetail**  
  Retorna as informações de registro para uma entidade de negócio, incluindo todos os serviços que essa entidade fornece.
* **get\_businessDetailExt**  
  Retorna as informações completas de registro para uma entidade de negócio.
* **get\_serviceDetail**  
  Retorna as informações completas de registro para um serviço web.
* **get\_TModelDetail**  
  Retorna as informações completas de registro para um TModel.

O **InquireSOAP** define a interface de serviço web para pesquisar o registro UDDI. O Exemplo 6-8 mostra as definições dos métodos **find\_binding**, **find\_business** e **find\_service**.

**Exemplo 6-8. InquireSOAP**



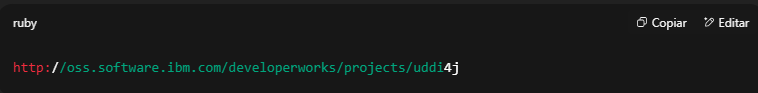
As definições de mensagens são tão simples quanto na interface Publisher. O Exemplo 6-9 mostra as três primeiras. Consulte o Apêndice C para a lista completa.

**Exemplo 6-9. Definições de mensagens Inquiry**



**6.3 Usando UDDI para Publicar Serviços**

Existem vários toolkits, tanto de código aberto quanto de código fechado, que fornecem uma implementação das interfaces **Publish** e **Inquiry** do UDDI. Vamos guiá-lo no uso de um pacote de código aberto da IBM chamado **UDDI4J** (UDDI for Java). Você pode baixar este pacote em:



Os passos para usar o UDDI4J para publicar serviços web são:

1. Registrar o provedor de serviços como uma entidade de negócio UDDI.
2. Especificar as categorias e identificadores que se aplicam à sua entrada de entidade de negócio.
3. Registrar o serviço web como um serviço de negócio UDDI.
4. Especificar as categorias que se aplicam à sua entrada de serviço de negócio.
5. Registrar os detalhes de implementação do seu serviço web, incluindo a localização na rede onde o serviço está implantado.

O modelo de dados UDDI nos permite realizar todos esses passos em uma única operação.

**6.3.1 Programa de Registro**

Um programa Java para publicar o serviço Hello World é fornecido no Apêndice C. Vamos analisar os destaques, que demonstram como usar o toolkit UDDI4J.

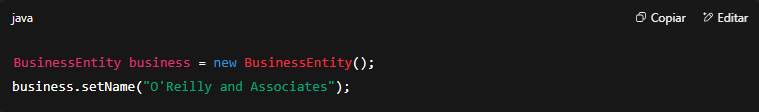
Você usa o UDDI4J por meio de um objeto proxy, que lida com a codificação e decodificação SOAP subjacente. Você deve inicializar o proxy para o registro UDDI conforme mostrado no Exemplo 6-10.

**Exemplo 6-10. Inicializando o Proxy UDDI**



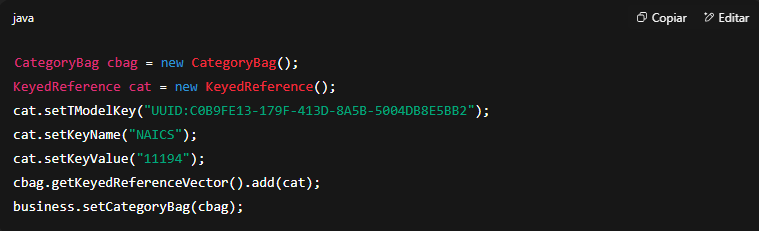
O UDDI4J define classes para os tipos de dados UDDI. Elas possuem métodos acessores diretos, então você prepara o registro de entidade de negócio como no Exemplo 6-11.

**Exemplo 6-11. Especificando a entidade de negócio**



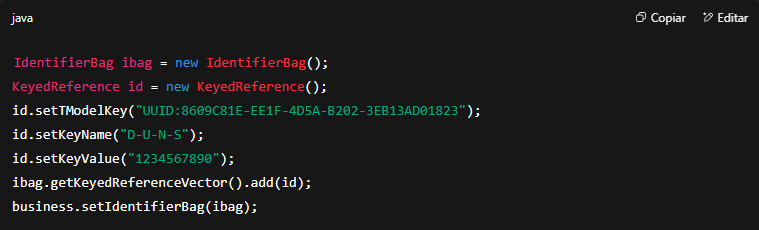
Da mesma forma, você pode especificar as categorias e identificadores para essa entidade de negócio. No Exemplo 6-12, usamos um código de categoria do **North American Industry Classification System (NAICS)** de 11194.

**Exemplo 6-12. Especificando categorias e identificadores para a entidade de negócio**



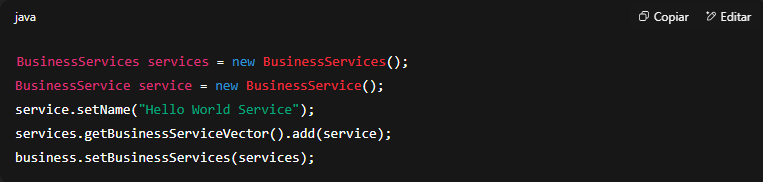
No Exemplo 6-13, preparamos os identificadores para a entidade de negócio. Especificamos um número **Dun and Bradstreet** que pode ser usado para identificar a entidade de negócio (é fictício, mas você entendeu a ideia). Como é possível ter mais de um identificador para uma empresa, o UDDI4J define uma classe **IdentifierBag** que mantém os identificadores individuais.

**Exemplo 6-13. Identificadores de entidade de negócio**



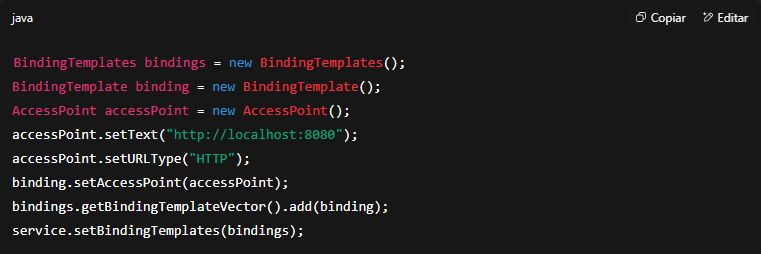
Prepare o registro de serviço de negócio como no Exemplo 6-14.

**Exemplo 6-14. Inicializando o registro de serviço de negócio**



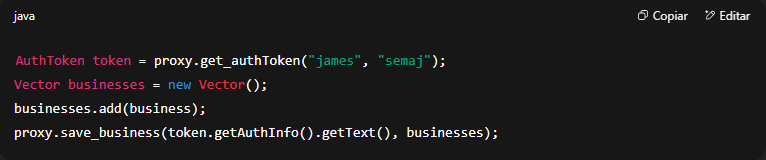
O Exemplo 6-15 mostra a inicialização dos modelos de ligação. O modelo de ligação especifica os protocolos implementados por um serviço e a localização na rede. É o equivalente no UDDI à definição de **binding** e **service port** no WSDL.

**Exemplo 6-15. Inicializando os modelos de ligação**



O Exemplo 6-16 realiza o login no registro UDDI e registra a entidade de negócio.

**Exemplo 6-16. Registrando a entidade de negócio**



**6.3.2 Como Registrar**

Você precisará de duas coisas antes de poder usar o programa de registro:

1. Você deve ter uma conta de usuário válida no registro UDDI que escolher. Você a adquire registrando-se por meio da interface em HTML fornecida pelo provedor específico de registro UDDI.
2. Você deve ter o **Apache SOAP** Versão 2.1 ou superior no seu *classpath* Java (o UDDI4J usa o Apache SOAP). Para atender a esse requisito, certifique-se de que **soap.jar**, **mail.jar** e **activation.jar** estejam todos no seu *classpath*.

Existem três situações comuns que causam erro ao registrar um serviço:

1. Uma empresa já pode existir com o nome especificado.
2. Pode haver algum problema com as informações definidas.
3. Você pode não ter permissões adequadas para executar a ação solicitada.

**6.3.3 O Envelope SOAP para o Registro**

O envelope SOAP enviado ao registro UDDI inclui todas as informações de registro para a entidade de negócio, como mostrado no Exemplo 6-17.

**Exemplo 6-17. Envelope SOAP para o registro**



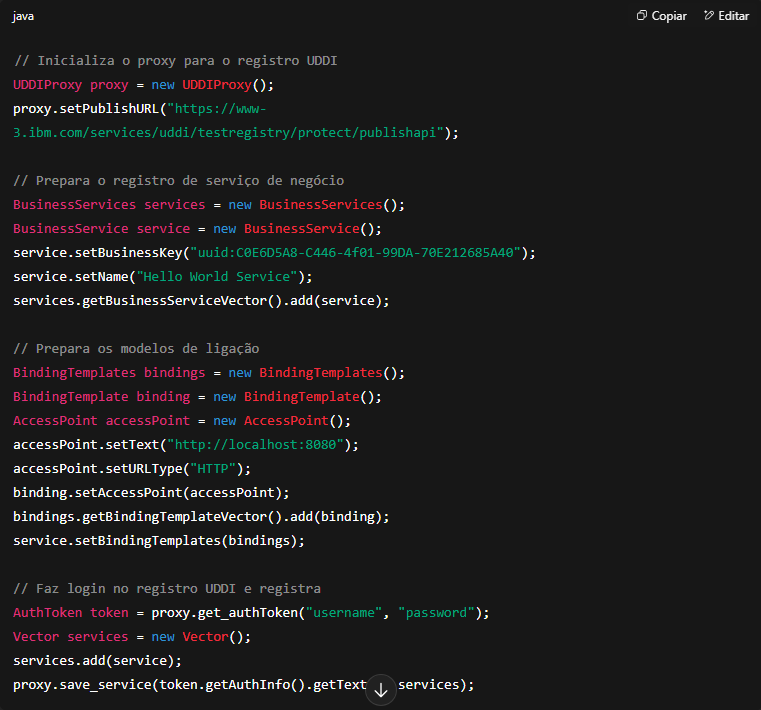
**6.3.4 Outros Assuntos**

Operações como **save\_business** são destrutivas. Em outras palavras, quando você instrui o registro UDDI a salvar uma entidade de negócio, o registro usará as informações fornecidas para substituir todas as outras informações dessa entidade de negócio existentes no registro. Há duas maneiras de contornar isso:

1. Recuperar o registro completo da entidade de negócio do registro UDDI antes de fazer qualquer alteração nas informações (por exemplo, ao publicar um novo serviço). Faça todas as alterações diretamente no registro recebido do registro UDDI. Salvar grava seu registro modificado.
2. Salvar apenas as partes específicas que você está alterando. Por exemplo, se você já possui um registro de entidade de negócio em um registro UDDI e tudo o que deseja fazer é registrar um novo serviço, então você deve usar a operação **save\_service** em vez de **save\_business**. Isso reduz a quantidade de dados que estão sendo transferidos e compartimentaliza as alterações que estão sendo feitas.

O Exemplo 6-18 usa **save\_service** para localizar as alterações.

**Exemplo 6-18. Alterando apenas alguns campos no registro**



A única diferença real é a ausência da entidade de negócio e a adição da linha **service.setBusinessKey**. Isso informa ao registro UDDI qual entidade de negócio deve ser atualizada. Este ID é gerado automaticamente pelo registro UDDI e retornado ao cliente quando o registro da entidade de negócio é criado.

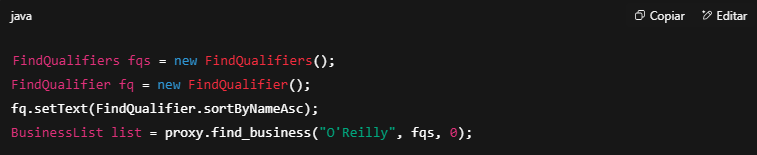
**6.4 Usando UDDI para Localizar Serviços**

O **UDDI4J** também pode ser usado para localizar serviços que foram publicados dentro de um registro UDDI. O processo envolve o uso de várias operações **find**, como **find\_business**, **find\_service** e **find\_binding**.

O Apêndice C contém um programa para pesquisar uma entidade de negócio e navegar nos resultados dessa operação para descobrir informações sobre os serviços fornecidos por essa entidade. Vamos discutir os principais pontos.

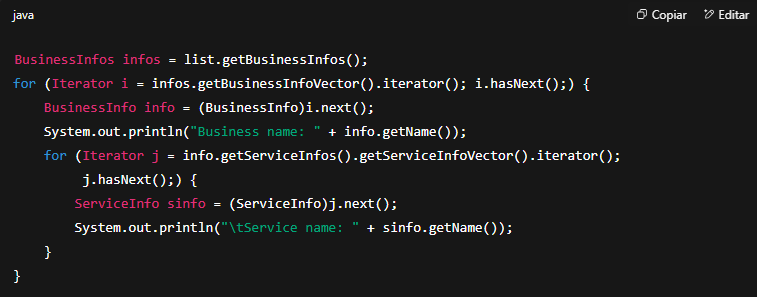
**FindQualifiers** modifica as operações de pesquisa indicando se a busca deve diferenciar maiúsculas e minúsculas (*case-sensitive*), se os resultados devem ser classificados em ordem ascendente ou descendente, e se a correspondência exata de nomes é necessária. O último argumento em todas as operações **find** é o número máximo de resultados a serem retornados. Passar o número zero indica que todos os resultados correspondentes devem ser retornados. O Exemplo 6-19 configura o **FindQualifiers** para procurar pelo negócio “O'Reilly”.

**Exemplo 6-19. FindQualifiers para procurar O'Reilly**

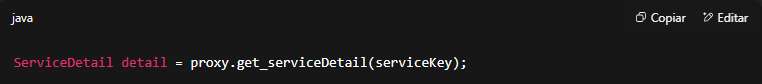


Entidades de negócio correspondentes são retornadas juntamente com uma listagem dos serviços oferecidos. A listagem inclui o nome e o identificador único do serviço. Use o UUID para aprofundar e obter mais informações sobre o serviço, como no Exemplo 6-20.

**Exemplo 6-20. Obtendo mais informações sobre o serviço**



Para recuperar informações mais específicas sobre um determinado serviço, use a operação **get\_serviceDetail** e passe o identificador único do serviço que você está solicitando:



Usando as informações contidas no detalhe do serviço, um cliente pode conectar-se e invocar o serviço web.

**6.5 Gerando UDDI a partir de WSDL**

Como existem algumas variações e sobreposições na forma como WSDL e UDDI dão suporte à descrição de serviços web, a coalizão da indústria que conduz o UDDI lançou um documento descrevendo as melhores práticas a serem seguidas ao usar UDDI e WSDL juntos para permitir a descoberta dinâmica de descrições de serviços web. Basicamente, ele define como usar uma descrição WSDL para gerar o registro UDDI de um serviço.

Primeiro, divida a descrição WSDL em duas partes (dois arquivos WSDL separados). O primeiro arquivo se torna a **descrição de interface**. Ele inclui os tipos de dados, mensagens, tipos de porta (*port types*) e vinculações (*bindings*). O segundo arquivo é conhecido como **descrição de implementação**. Ele inclui apenas a definição do serviço. A descrição de implementação importa a descrição de interface usando o mecanismo <wsdl:import />.

**6.5.1 Descrição de Interface**

O Exemplo 6-21 é a descrição de interface para o nosso exemplo Hello World.

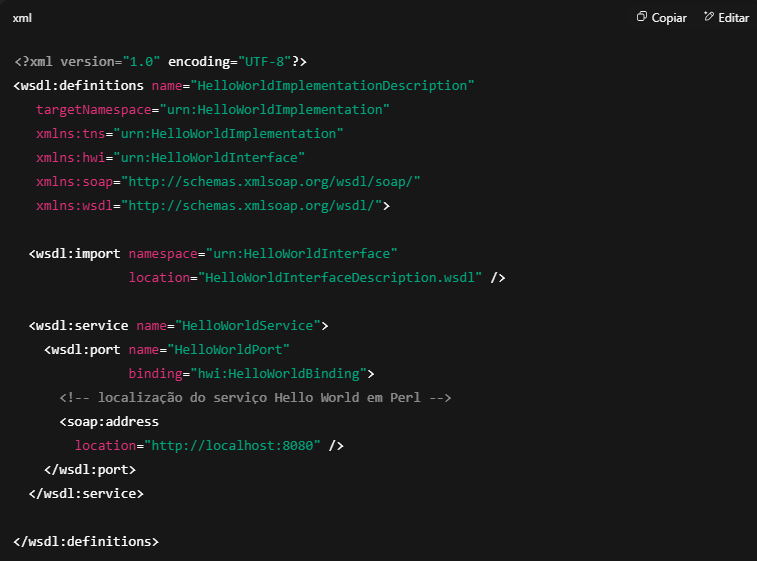
**Exemplo 6-21. HelloWorldInterfaceDescription.wsdl**



**6.5.2 Descrição de Implementação**

O Exemplo 6-22 é a descrição de implementação WSDL para o nosso exemplo Hello World.

**Exemplo 6-22. HelloWorldImplementationDescription.wsdl**



**6.5.3 Registro**

Registre a descrição de interface como um **TModel** do UDDI. Você já viu categorias NAICS e identificadores D-U-N-S como TModels. Outro tipo de TModel é uma descrição WSDL de uma interface de serviço.

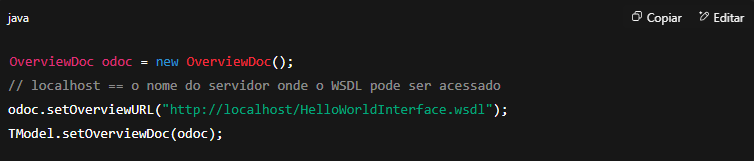
Para registrar a interface como um TModel, crie uma estrutura TModel e use a operação **save\_TModel**, como no Exemplo 6-23.

**Exemplo 6-23. Registrando a descrição de interface como um TModel**



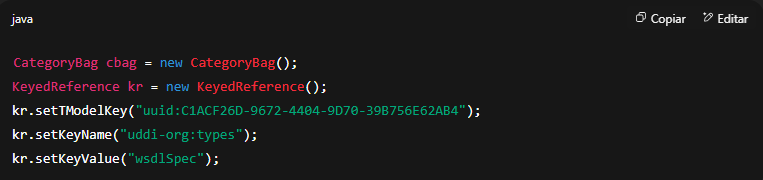
O **OverviewDoc** é um ponteiro para a descrição WSDL de interface, hospedada em um servidor web publicamente acessível. O Exemplo 6-24 mostra como configurar isso.

**Exemplo 6-24. Configurando o OverviewDoc**



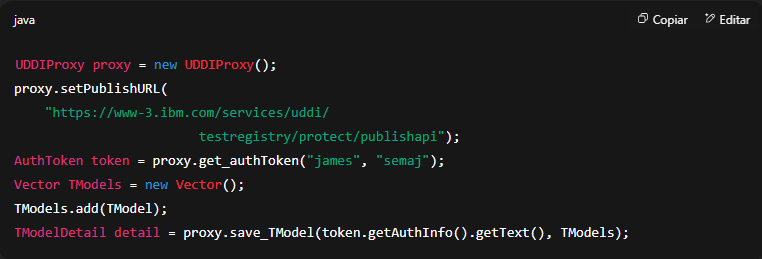
Indique que este TModel representa uma descrição de interface WSDL criando uma referência de categoria com um **TModelKey** de uuid:C1ACF26D-9672-4404-9D70-39B756E62AB4, um nome de chave (**keyName**) de uddi-org:types e um valor de chave (**keyValue**) de wsdlSpec, como mostrado no Exemplo 6-25.

**Exemplo 6-25. Marcando o TModel como WSDL**



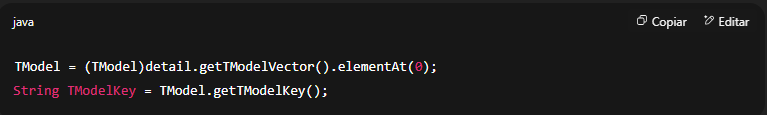
O Exemplo 6-26 mostra como chamar a operação **save\_TModel** para registrar o TModel.

**Exemplo 6-26. Chamando a operação save\_TModel**



A operação **save\_TModel** retorna uma cópia do registro TModel recém-registrado, incluindo o identificador único gerado automaticamente. Guardamos essa chave única para o próximo passo, como mostrado no Exemplo 6-27.

**Exemplo 6-27. Mantendo a chave única**



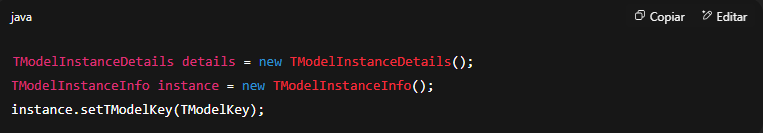
Agora, diga o que é o serviço e onde ele está localizado, como no Exemplo 6-28.

**Exemplo 6-28. Definindo um serviço, modelo de ligação e ponto de acesso para o serviço**



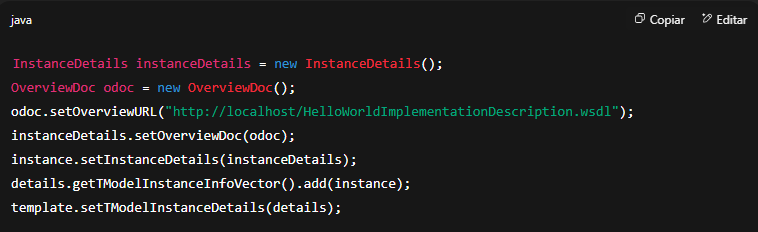
O Exemplo 6-29 especifica que este serviço é uma instância do TModel **HelloWorld-InterfaceDescription** recém-registrado. A variável **TModelKey** é o identificador único obtido no Exemplo 6-27.

**Exemplo 6-29. Associando o serviço ao TModel**



Forneça um link para a descrição WSDL de implementação como no Exemplo 6-30. Assim como a descrição de interface, ela precisa estar localizada em algum endereço web publicamente acessível.

**Exemplo 6-30. Ligando à descrição WSDL de implementação**



Uma vez preparado o registro, inicialize o proxy e chame a operação **save\_service** para registrar o serviço de negócio. O Exemplo 6-31 mostra isso, de forma abreviada. Veja o Apêndice C para o código completo.

**Exemplo 6-31. Salvando as informações do serviço**



Seguindo essas diretrizes, WSDL e UDDI podem trabalhar muito bem juntos.

**6.6 Usando UDDI e WSDL Juntos**

Uma vez que o serviço web definido em WSDL é publicado em um registro UDDI, é possível criar *proxies* de serviço altamente dinâmicos. O **IBM Web Services ToolKit**, por exemplo, fornece suporte integrado para localizar serviços no UDDI e invocar esses serviços por meio de um *proxy* dinâmico configurado a partir do WSDL.

Para mostrar mais do que está acontecendo nos bastidores, no entanto, vamos usar o **UDDI4J** e o **WSIF** juntos para implementar o mesmo tipo de funcionalidade.

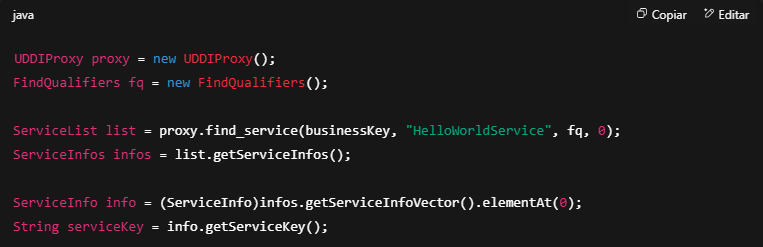
Os passos são simples:

1. Localizar o serviço Hello World no registro UDDI.
2. Acessar a descrição WSDL para o serviço Hello World.
3. Invocar o serviço Hello World.

Tudo isso é feito no lado do cliente. Nada precisa ser feito no servidor para que isso funcione.

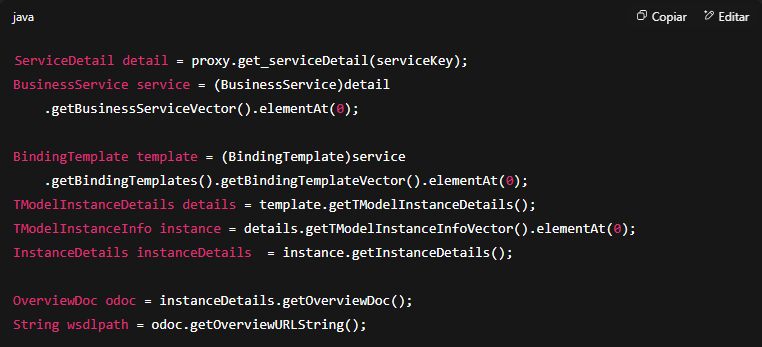
Primeiro, escreva o código para localizar o serviço Hello World no UDDI. O Exemplo 6-32 pesquisa com **FindQualifiers** e pega o primeiro resultado retornado pelo servidor UDDI.

**Exemplo 6-32. Localizando um serviço Hello World**



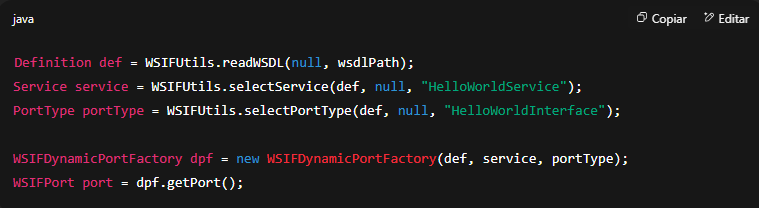
Com o identificador único do serviço correspondente, o Exemplo 6-33 vai ao registro UDDI para recuperar o serviço de negócio. O modelo de ligação (*binding template*) para esse serviço então identifica a implementação a ser usada.

**Exemplo 6-33. Localizando uma implementação**



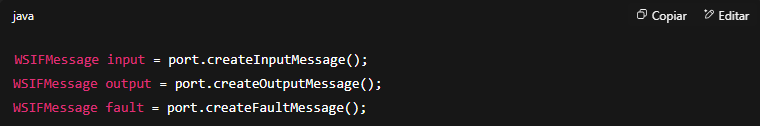
De posse do WSDL, o Exemplo 6-34 usa o **WSIF** para invocar o serviço web.

**Exemplo 6-34. Invocando o serviço web com WSIF**



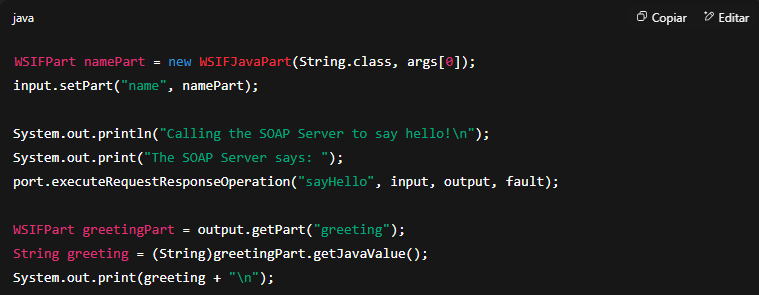
Normalmente, a criação de mensagens é feita nos bastidores, fora da visão do programador. O Exemplo 6-35 mostra isso.

**Exemplo 6-35. Criando mensagens com WSIF**



O Exemplo 6-36 chama o serviço Hello World.

**Exemplo 6-36. Invocando Hello World**



A execução disso produz a mesma saída que vimos nos outros exemplos de serviços Hello World (mostrado no Exemplo 6-37).

**Exemplo 6-37. Saída do cliente Hello World usando WSDL, UDDI e WSIF**



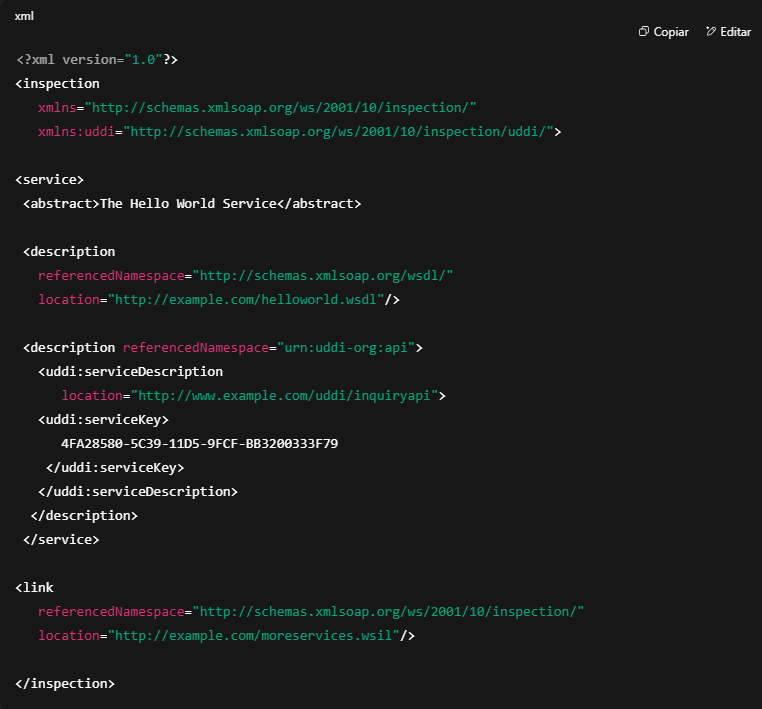
O programa descobriu dinamicamente, inspecionou e vinculou-se ao serviço web Hello World. Nós não programamos o cliente sabendo qual implementação usaríamos. Embora o cliente esteja em Java, não há razão para que não pudesse ter sido escrito em qualquer linguagem. **C#**, **Visual Basic** e **Perl** também possuem extensões para UDDI e WSDL.

**6.7 A Linguagem de Inspeção de Serviços Web (WS-Inspection)**

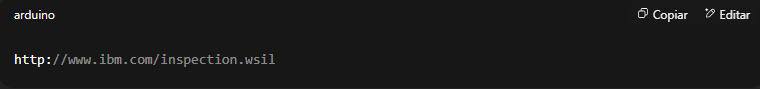
Embora o UDDI seja o mecanismo mais conhecido para descoberta de serviços, ele não é nem o único mecanismo nem sempre a melhor ferramenta para o trabalho. Em muitos casos, a complexidade e o escopo do UDDI são exagerados se tudo o que é necessário for um simples ponteiro para um documento WSDL ou um *endpoint* de URL de serviços. Reconhecendo isso, a IBM e a Microsoft se reuniram e desenvolveram uma proposta para uma nova **Linguagem de Inspeção de Serviços Web** que pode ser usada para criar um índice simples de descrições de serviços em um determinado local de rede.

Um exemplo de documento WS-Inspection é ilustrado no Exemplo 6-38. Ele contém uma referência a um único serviço (Hello World) com duas descrições — uma descrição baseada em WSDL e uma descrição baseada em UDDI.

**Exemplo 6-38. Um documento WS-Inspection simples**



Uma vez criados, os documentos WS-Inspection devem ser colocados em um local bem conhecido ou facilmente descoberto no seu servidor web. De fato, a especificação WS-Inspection define que, no mínimo, um documento de inspeção chamado **inspection.wsil** deve estar disponível na raiz do servidor: por exemplo,



Isso permite que clientes potenciais desses serviços localizem facilmente os documentos de inspeção e, assim, descubram os serviços que estão sendo anunciados.

A relação entre UDDI e WS-Inspection é simples. O UDDI é como uma lista telefônica. Se você precisa de um encanador para consertar os canos sob sua pia, mas não conhece um bom para ligar, você abre a lista telefônica e encontra um. Se você precisa de um serviço web que implemente um determinado *port type* WSDL para processar pedidos de compra de rolamentos de esferas, pode enviar uma solicitação a um registro UDDI para encontrar um serviço apropriado.

O WS-Inspection, no entanto, é útil se você já conhece o provedor de serviços que deseja usar (por exemplo, você já sabe qual encanador vai chamar, então não precisa olhar na lista telefônica). Você simplesmente consultaria o documento WS-Inspection publicado pelo provedor de serviços para encontrar a localização dos serviços que ele oferece.

**6.7.1 Sintaxe do WS-Inspection**

A sintaxe de um documento WS-Inspection é simples. O elemento raiz **inspection** contém uma coleção de elementos **abstract**, **link** e **service**. O elemento **abstract** fornece documentação simples ao longo do documento WS-Inspection. O elemento **link** permite que o documento de inspeção se vincule a outros documentos de inspeção externos ou até mesmo a outros mecanismos de descoberta (como um registro UDDI) onde informações adicionais podem ser encontradas.

O elemento **service** representa um serviço web sendo oferecido pelo publicador do documento de inspeção.

O elemento **service** em si é uma coleção de elementos **abstract** e **description**. Você pode descrever um serviço de várias maneiras. O WS-Inspection permite que todas as descrições de um serviço sejam listadas. É possível fornecer informações estendidas sobre cada descrição de serviço usando a extensibilidade do XML.

O Exemplo 6-38, por exemplo, contém tanto uma descrição baseada em WSDL quanto uma baseada em UDDI.

O WS-Inspection será submetido para padronização em algum momento. Por enquanto, tanto a IBM quanto a Microsoft implementaram suporte para ele em suas ofertas de serviços web, e outros fornecedores de *toolkits* de serviços web estão considerando fazer o mesmo. Devido à sua utilidade e sintaxe simples, o WS-Inspection provavelmente desenvolverá um suporte favorável.

**Tópicos deste capítulo:**

* 6.1 O Registro UDDI
* 6.2 As Interfaces do UDDI
* 6.3 Usando UDDI para Publicar Serviços
* 6.4 Usando UDDI para Localizar Serviços
* 6.5 Gerando UDDI a partir de WSDL
* 6.6 Usando UDDI e WSDL Juntos
* 6.7 A Linguagem de Inspeção de Serviços Web (WS-Inspection)