Web Services

- XML
- SOAP
- WSDL
- UDDI
- Desenvolvimento
- Informações Adicionais

Web Services

- Definição
  - "Web services são aplicações modulares auto-descritas e auto-contidas, que podem ser conectadas e acopladas a outros web services" [IBM]
  - "Web Services são componentes de software com baixo fator de acoplamento, utilizados por meio de padrões de tecnologia Internet. Um Web Service representa uma função de negócio ou um serviço que pode ser acessado por uma outra aplicação." [Gartner]

Web Services

- Principais Características
  - Fornecem serviços a clientes dispersos na Web
  - Podem ser facilmente localizados na rede
  - Interfaces são bem definidas e auto-descritas
  - Empregam padrões da Internet
    - Formato de dados
    - Comunicação
  - Se comunicam facilmente através de *firewalls*
  - Consiste em uma tecnologia aberta, independente de linguagem e de plataforma

Web Services

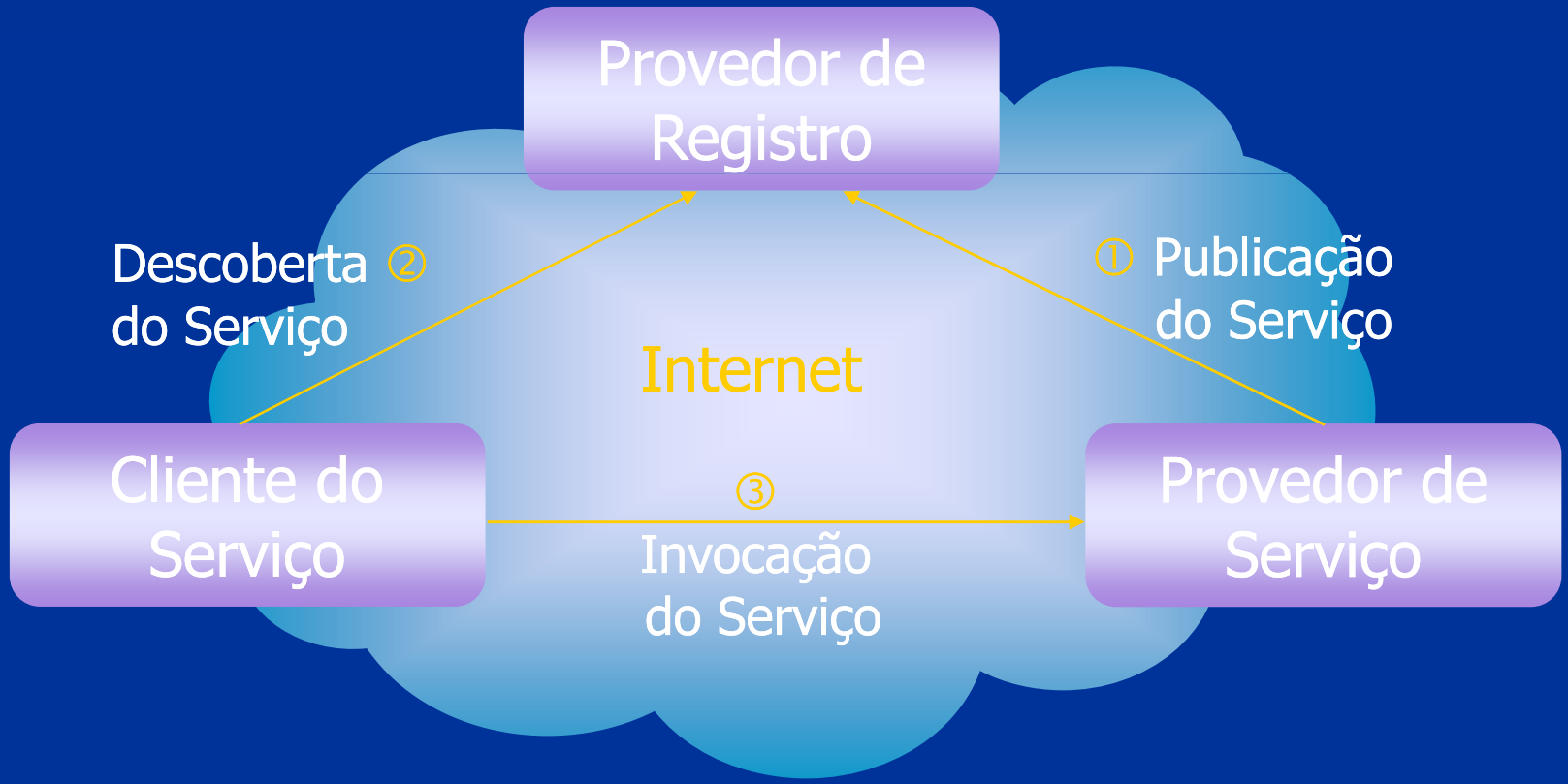
- Utilização
  - Construção de aplicações distribuídas baseadas na Web
  - Disponibilização de serviços pela Web
  - Integração de sistemas: PCs, dispositivos móveis, servidores de bancos de dados, ...
  - Implementação de regras de negócio no servidor Web
  - Gerenciamento de transações distribuídas em aplicações Web

Web Services

- Arquitetura Orientada a Serviços (SOA)
  - Provedor do serviço (*service provider*)
  - Provedor de registro (*registry provider*)
  - Cliente do serviço (*service requestor*)
- Interação entre os elementos
  - Publicação de serviços:
    provedor de serviço com provedor de registro
  - Descoberta de serviços:
    cliente com provedor de registro
  - Invocação de serviços:
    cliente com provedor de serviço

Web Services

- Elementos da Arquitetura



Web Services

- Comparação com outras tecnologias para desenvolvimento de aplicações distribuídas
  - CORBA
    - Solução aberta
    - Permite implementação do cliente e servidor em qualquer linguagem
    - Emprega IDL para descrever a interface
    - Formato de dados próprio
    - Grande complexidade de desenvolvimento
    - Geralmente apresenta bom desempenho

Web Services

- Comparação com outras tecnologias (cont.)
  - Java RMI
    - Comunicação apenas entre aplicações Java (a não ser que use RMI/IIOP, permitindo a comunicação também com objetos CORBA)
    - Independência de plataforma (JVM)
    - Dispensa o uso de IDL – API de reflexão do Java é usada para inspecionar as interfaces
    - Fácil desenvolvimento de aplicações
    - Desempenho limitado pelo uso do Java

Web Services

- Comparação com outras tecnologias (cont.)
  - Microsoft COM, DCOM e ActiveX
    - Suporte limitado quase que somente ao sistema operacional Windows
    - Protocolo de comunicação e formato de dados proprietários da Microsoft
    - Aplicações podem ser desenvolvidas em várias linguagens
    - Interfaces descritas em Microsoft IDL
    - Bom desempenho

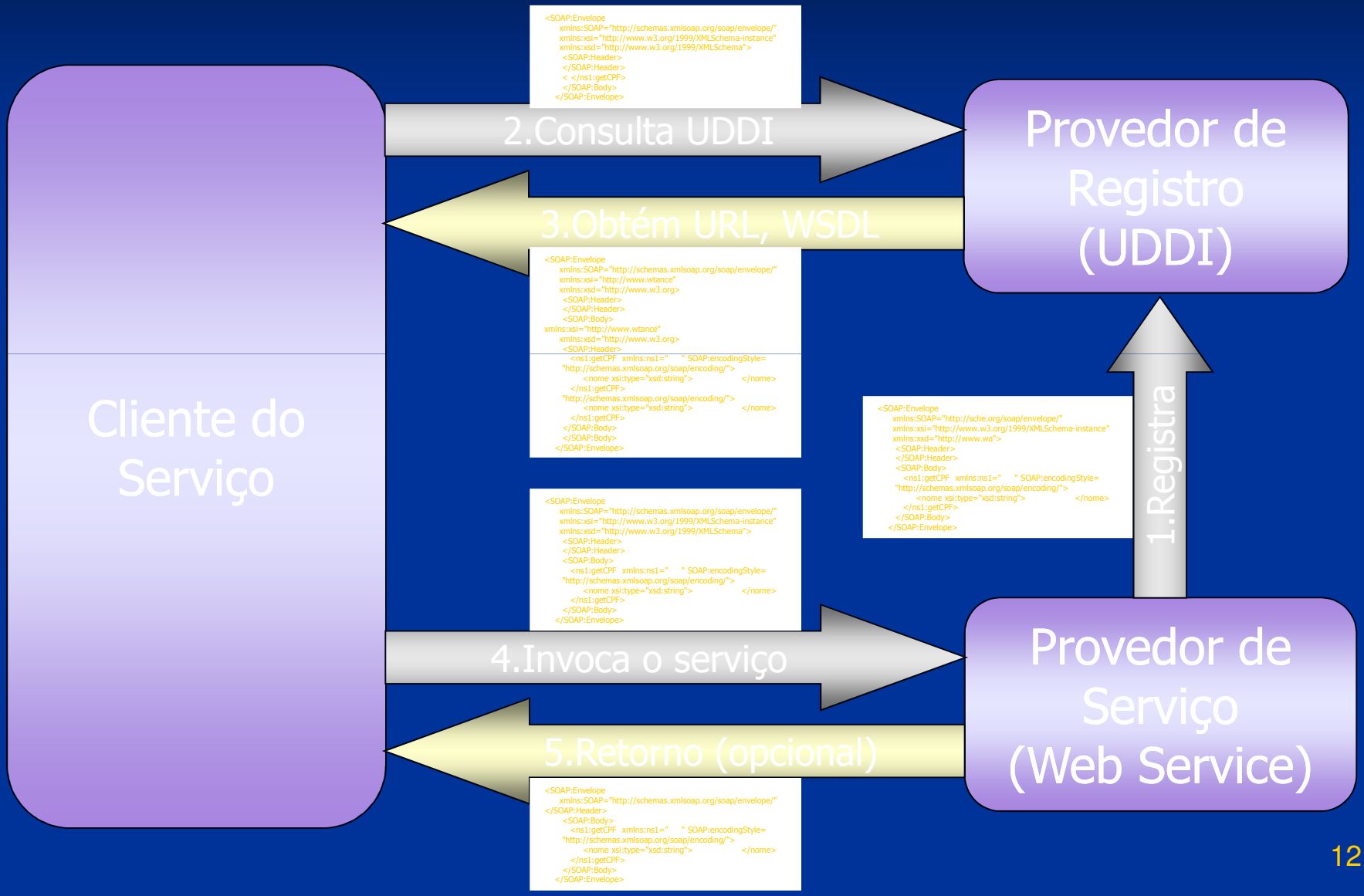
Web Services

- Comparação com outras tecnologias (cont.)
  - Web Services x Outros
    - Baseados em padrões abertos de grande aceitação no mercado
    - Protocolo de comunicação e formato de dados adotam padrões da Web
    - Infra-estrutura já disponível
    - Aplicações podem ser desenvolvidas facilmente usando qualquer linguagem
    - Interfaces claramente especificadas
    - Desempenho limitado

Web Services

- Tecnologias empregadas por Web Services
  - XML (*eXtensible Markup Language*): formato padrão para troca de dados
  - SOAP: protocolo utilizado na interação com os serviços Web
  - WSDL (*Web Services Description Language*): utilizada para descrever os serviços Web
  - UDDI (*Universal Description, Discovery and Integration*): permite localizar serviços na rede

Web Services



XML

- XML é uma linguagem extensível de marcação de dados definida pelo W3C
- XML é usada para intercambiar dados
  - Permite trocar dados facilmente entre aplicações Web
  - Facilita a análise de dados por programas
  - É independente de sistemas operacionais ou formatos proprietários usados por aplicações
  - Permitindo a definição de elementos pelo usuário (ou aplicação) para estruturar dados

XML

- Documentos XML
  - Documentos estruturados em formato texto
  - Compostos por tags XML e valores dos dados
  - Tags podem ser definidas pelo usuário
  - Legíveis para humanos e máquinas
  - Os dados contidos em um documento XML podem ser facilmente interpretados pelas aplicações, independentemente de linguagem de desenvolvimento, do sistema operacional e do protocolo de comunicação utilizado

XML

- Documentos XML x HTML
  - XML é visto erroneamente como um formato alternativo ao HTML
  - XML não possui tags para formatação de documentos, como o HTML
  - XML se preocupa apenas com o conteúdo do documento, e não com a sua apresentação

XML

- Apresentação de documentos XML
  - Os dados de um documento XML podem ser apresentados de várias maneiras, dependendo do contexto no qual são utilizados
  - Folhas de estilo XSL (*eXtensible Stylesheet Language*) especificam regras para apresentar um documento XML (em HTML, PDF, ...)
  - Diferentes folhas de estilo podem ser aplicadas a um mesmo documento XML, apresentando o dado de forma diferente em cada situação ou para diferentes usuários

XML

- Elementos de um documento XML
  - Especificados usando *tags*
    - Tag de abertura: <tag>
    - Tag de fechamento: </tag>
    - Tag com auto-fechamento: <tag />
  - Os valores dos dados são especificados entre tags de abertura e fechamento: <tag>dado</tag>
  - Tags podem possuir atributos: <tag atrib="valor" />
  - Tags podem conter outras tags aninhadas: <tag1> <tag2>dado</tag2> </tag1>

XML

- Documentos XML devem ser bem-formados
  - Devem conter apenas um elemento, que é a raiz da árvore XML
  - O elemento raiz pode conter outros elementos
  - Todos os elementos especificados em XML devem ser finalizados, ao contrário de HTML
  - Elementos aninhados devem ser finalizados na ordem inversa de abertura

XML

- Exemplo de Documento XML

```
<?xml version="1.0" encoding="UTF-8"?>
<previsaoDoTempo data="01/07/2008" unidade="Celsius">
  <localidade id="1" nome="Florianópolis">
    <TemperaturaMinima valor="14.5"/>
    <TemperaturaMaxima valor="21.3"/>
  </localidade>
  <localidade id="2" nome="São Joaquim">
    <TemperaturaMinima valor="7.2"/>
    <TemperaturaMaxima valor="15.1"/>
  </localidade>
</previsaoDoTempo>
```

XML

- Esquemas XML
  - Especificam o formato que deve ser respeitado por um documento XML
  - Definem tags, atributos e os tipos de dados aceitos para cada elemento
  - Um documento XML é válido se estiver em conformidade com um esquema
  - Tipos de esquemas XML
    - DTD (*Document Type Definition*)
    - XSD (*XML Schema Definition*)

XML

- DTD
  - Formato não-XML
  - Pode ser embutido no XML ou especificado em um arquivo em separado (extensão .dtd)
  - Especifica os elementos aceitos, seus atributos e os elementos que este pode conter
  - Limitação: não define os tipos de dados e os valores aceitos em cada campo do documento

XML

- Exemplo de DTD

```
<!ELEMENT Temps (localidade*)>
<!ATTLIST previsaoDoTempo data CDATA #REQUIRED>
<!ATTLIST previsaoDoTempo unidade CDATA #REQUIRED>
<!ELEMENT localidade (TemperaturaMinima,
                      TemperaturaMaxima)>
<!ATTLIST localidade id CDATA #REQUIRED>
<!ATTLIST localidade nome CDATA #REQUIRED>
<!ELEMENT TemperaturaMinima EMPTY>
<!ATTLIST TemperaturaMinima valor CDATA #REQUIRED>
<!ELEMENT TemperaturaMaxima EMPTY>
<!ATTLIST TemperaturaMaxima valor CDATA #REQUIRED>
```

XML

- XSD
  - Formato XML
  - Permite especificar os tipos de dados, o formato e os valores aceitos em cada campo
  - Pode ser facilmente reutilizado em outros esquemas através da definição de *namespaces*
  - Proposto pela *Microsoft* e posteriormente aceito como um padrão W3C

XML

- Exemplo de XSD

```
<?xml version="1.0" encoding="UTF-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="previsaoDoTempo" type="PrevisaoType"/>

<xs:complexType name="PrevisaoType">
  <xs:sequence>
    <xs:element name="localidade" type="LocalidadeType"
                minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="data" type="xs:string" use="required"/>
  <xs:attribute name="unidade" type="xs:string" use="required"/>
</xs:complexType>

+ <xs:complexType name="LocalidadeType">

</xs:schema>
```

XML

- Parsers
  - São responsáveis por fazer a verificação de um documento XML, obtendo os dados que serão usados por uma determinada aplicação
- APIs
  - Usadas para fazer o parsing de documentos
  - Exemplos de APIs XML:
    - DOM
    - SAX

SOAP

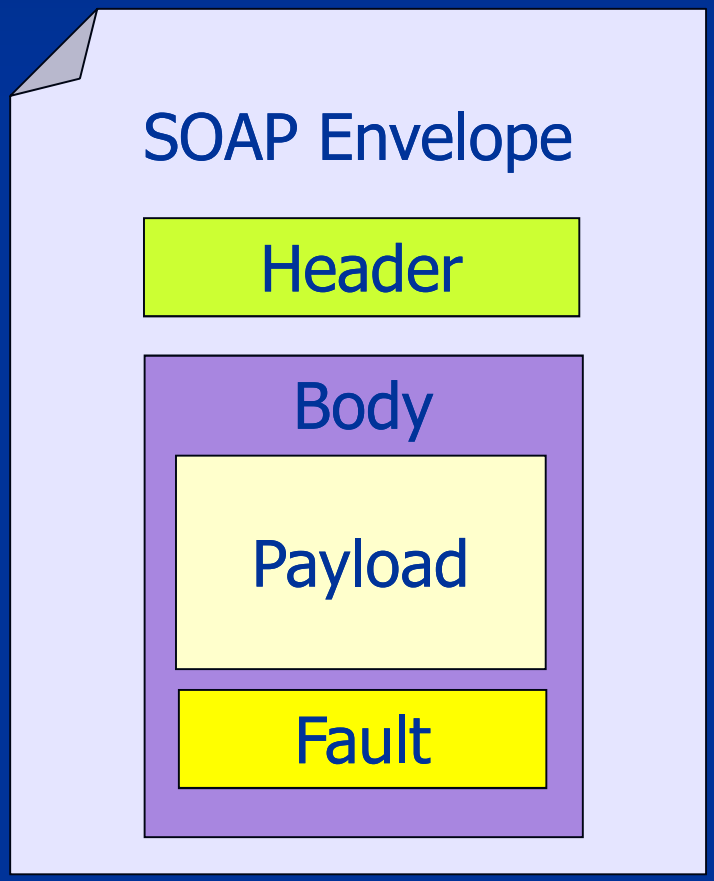
- Protocolo SOAP
  - Protocolo definido pelo W3C para comunicação entre Web Services
  - Nome originou das iniciais de *Simple Object Access Protocol* (esse nome não é mais usado)
  - Define o formato das mensagens trocadas entre Web Services
  - Independente de plataforma e de linguagem
  - Utiliza em geral HTTP[S] como protocolo de transporte (porta 80 → atravessa *firewalls*)

SOAP

- Funcionamento
  - Cliente cria um envelope SOAP especificando o nome da operação requisitada e os nomes e valores dos parâmetros da operação
  - Requisição é enviada pela rede ao provedor do serviço
  - Requisição é recebida e interpretada
  - A operação requisitada é executada
  - A resposta, se houver, é colocada em um envelope SOAP e enviada ao cliente

SOAP

- Envelope SOAP



SOAP

- Exemplo de Requisição SOAP

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
<S:Header/>
<S:Body>
   <ns1:getTemperaturaMinima xmlns:ns1="http://ufsc.br/previsao">
      <localidade>Florianópolis</localidade>
   </ns1:getTemperaturaMinima>
</S:Body>
</S:Envelope>
```

SOAP

- Exemplo de Resposta SOAP

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
<S:Body>
   <ns1:getTemperaturaMinimaResponse
     xmlns:ns1="http://ufsc.br/previsao">
      <return>13.2</return>
   </ns1:getTemperaturaMinimaResponse>
</S:Body>
</S:Envelope>
```

WSDL

- Linguagem de descrição de Web Services
  - Padrão do W3C
  - Baseado no XML
  - Especifica a interface de um serviço Web
  - Através do WSDL de um Web Service é possível saber que serviços estão disponíveis e como invocá-los remotamente
  - A especificação WSDL é independente da linguagem na qual o Web Service é implementado
  - Equivalente à especificação de interface IDL de um objeto CORBA ou DCOM

WSDL

- Estrutura

```
<?xml version="1.0" encoding="UTF-8"?>
 <definitions name="PrevisaoDoTempo"
                targetNamespace="http://ufsc.br/previsao"
                xmlns:tns="http://ufsc.br/previsao"
                xmlns:xsd="http://www.w3.org/2001/XMLSchema"
                xmlns:soap="http://www.w3.org/2003/11/wsdl/soap12"
                xmlns="http://www.w3.org/2003/11/wsdl">
+ <types>
+ <message name="getTemperaturaMinima">
+ <message name="getTemperaturaMinimaResponse">
+ <message name="getTemperaturaMaxima">
+ <message name="getTemperaturaMaximaResponse">
+ <portType name="Tempo">
+ <binding name="TempoPortBinding" type="tns:Tempo">
+ <service name="TempoService">
  </definitions>
```

WSDL

- Elementos
  - <definitions>: elemento raiz
  - <types>: define os tipos de dados utilizados pelo serviço Web (pode referenciar um XSD)
  - <messages>: especifica as mensagens usadas na comunicação com o serviço Web
  - <portType>: define um conjunto de operações que são executadas por um serviço
  - <binding>: associa um protocolo ao serviço
  - <service>: especifica o endereço de rede no qual o serviço pode ser acessado

WSDL

- Definição de Tipos
  - Importa um XSD com a descrição dos tipos

```
<types>
  <xsd:schema>
    <xsd:import namespace="http://ufsc.br/previsao"
     schemaLocation="http://ufsc.br/previsao/tempo.xsd" />
  </xsd:schema>
</types>
```

WSDL

- Definição de Tipos em um XSD

```
<xs:schema xmlns:tns="http://ufsc.br/previsao"
                xmlns:xs="http://www.w3.org/2001/XMLSchema"
                version="1.0" targetNamespace="http://ufsc.br/previsao">
<xs:element name="getTemperaturaMinima"
                type="tns:getTemperaturaMinima" />
<xs:element name="getTemperaturaMinimaResponse"
                type="tns:getTemperaturaMinimaResponse" />
<xs:complexType name="getTemperaturaMinima">
  <xs:sequence>
    <xs:element name="localidade" type="xs:string" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="getTemperaturaMinimaResponse">
  <xs:sequence>
    <xs:element name="return" type="xs:float" />
  </xs:sequence>
</xs:complexType>
...
</xs:schema>
```

WSDL

- Definição de Mensagens

```
<message name="getTemperaturaMinima">
   <part name="parameters" element="tns:getTemperaturaMinima" />
</message>
<message name="getTemperaturaMinimaResponse">
   <part name="parameters"
         element="tns:getTemperaturaMinimaResponse"/>
</message>
<message name="getTemperaturaMaxima">
   <part name="parameters" element="tns:getTemperaturaMaxima" />
</message>
<message name="getTemperaturaMaximaResponse">
   <part name="parameters"
         element="tns:getTemperaturaMaximaResponse"/>
</message>
```

WSDL

- Definição de Porta

```
<portType name="Tempo">
  <operation name="getTemperaturaMinima">
    <input message="tns:getTemperaturaMinima" />
    <output message="tns:getTemperaturaMinimaResponse" />
  </operation>
  <operation name="getTemperaturaMaxima">
    <input message="tns:getTemperaturaMaxima" />
    <output message="tns:getTemperaturaMaximaResponse" />
  </operation>
</portType>
```

WSDL

- *Binding* com o Protocolo SOAP

```
<binding name="TempoPortBinding" type="tns:Tempo">
  <soap:binding transport="http://schemas.xmlsoap.org/soap/http"
   style="document" />
  <operation name="getTemperaturaMinima">
    <soap:operation soapAction="" />
      <input> <soap:body /> </input>
      <output> <soap:body /> </output>
  </operation>
  <operation name="getTemperaturaMaxima">
    <soap:operation soapAction="" />
      <input> <soap:body /> </input>
      <output> <soap:body /> </output>
  </operation>
</binding>
```

WSDL

- Definição de Serviço

```
<service name="TempoService">
  <documentation>Serviço de Previsão do Tempo</documentation>
  <port name="TempoPort" binding="tns:TempoPortBinding">
    <soap:address location="http://ufsc.br/previsao/TempoService" />
  </port>
</service>
```

UDDI

- Infra-estrutura para registro e localização de serviços Web
  - Padrão do OASIS que define um provedor de registros de Web Services
  - Criado por Ariba, IBM e Microsoft
  - Armazena as especificações WSDL dos provedores de serviços
  - Permite que os clientes encontrem os provedores dos serviços dos quais necessitam e descubram como solicitar tais serviços
  - Faz o papel do registro do RMI e dos serviços de nomes e *trading* do CORBA

UDDI

- Características
  - Repositório centralizado e universal para registro de serviços
  - Interfaces registradas são descritas em WSDL
  - Registros são armazenados em XML
  - Recebe requisições de registro e descoberta utilizando o protocolo SOAP
  - Empresas podem ter seus servidores UDDI privados para registro de serviços internos

UDDI

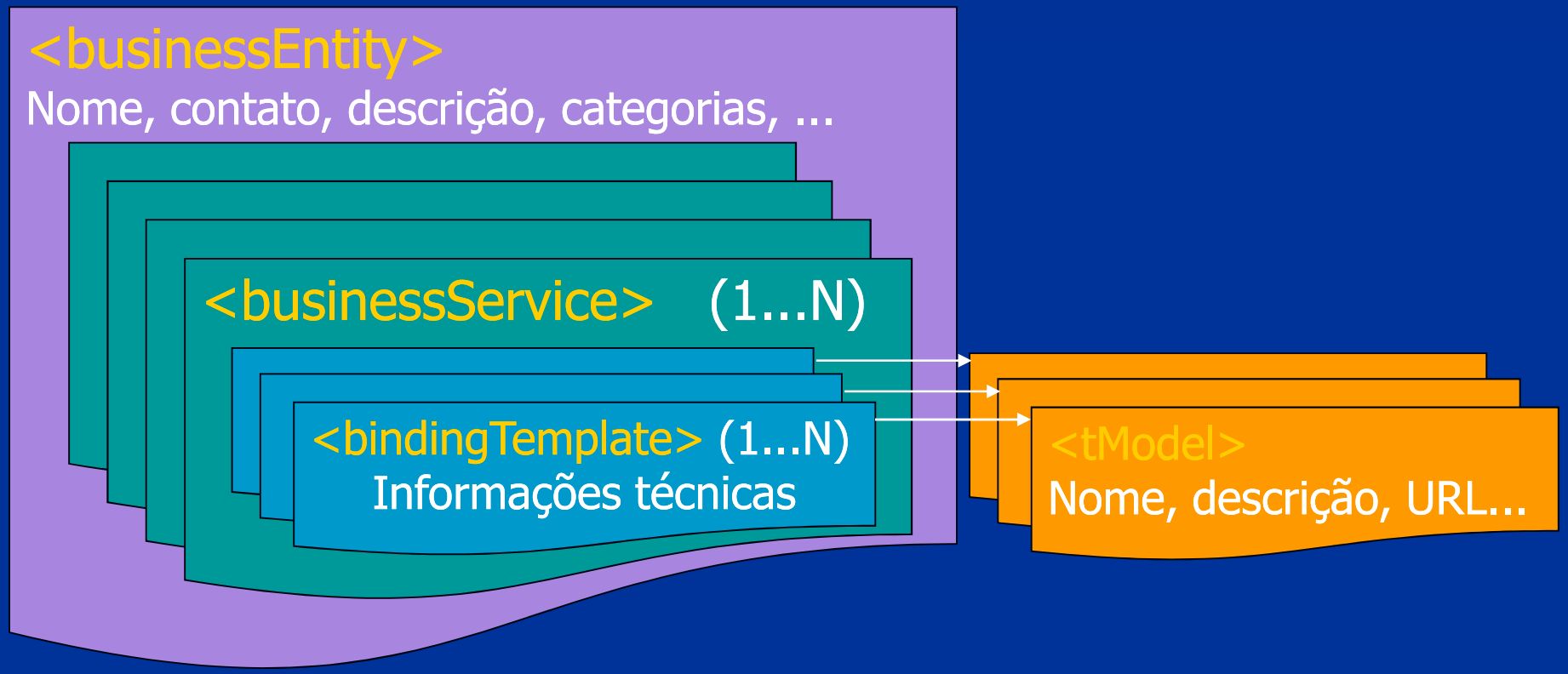
- Consulta de Registros
  - 
    Páginas Brancas
    - Fornecem o endereço para contato do provedor do serviço
  - 
    Páginas Amarelas
    - Classificam os provedores em categorias de acordo com o seu ramo de negócio
  - 
    Páginas Verdes
    - Fornecem informações técnicas sobre os serviços executados pelos provedores

UDDI

- Elementos do Registro UDDI
  - <businessEntity>: fornece informações sobre uma família de serviços
  - <businessService>: provê informações sobre um determinado serviço
  - <bindingTemplate>: contém informações técnicas referentes a um serviço
  - <tModel>: fornece informações específicas relacionadas ao serviço

UDDI

- Registro UDDI



UDDI

- UDDI possui APIs para:
  - Localização de Serviços
  - Publicação de Serviços
- Rotinas da API de Localização
  - find_binding
  - find_business
  - find_service
  - find_tModel
  - get_bindingDetail
  - get_businessDetail
  - get_serviceDetail
  - get_tModelDetail

UDDI

- Rotinas da API de Publicação
  - delete_binding
  - delete_business
  - delete_service
  - delete_tModel
  - save_binding
  - save_business
  - save_service
  - save_tModel

Desenvolvimento

- Toolkits facilitam a criação de Web Services
  - Podem gerar automaticamente:
    - Proxies (stubs/skeletons): processam as mensagens SOAP
    - WSDL do serviço web
    - Código para acesso ao UDDI
  - Exemplos:
    - Apache Axis
    - gSOAP
    - kSOAP
    - ...

Desenvolvimento

- APIs para criação de Web Services
  - Fornecem rotinas/classes para facilitar a manipulação de mensagens SOAP, para criar/interpretar descrições de serviços em WSDL e para acesso/registro no UDDI
  - APIs disponíveis no Java:
    - JAX-WS (*Java API for XML Web Services*)
    - JAXP (*Java API for XML Processing*)
    - JAXB (*Java API for XML Binding*)
    - JAX-RPC (*Java API for XML RPC*)
    - SAAJ (*SOAP with Attachments API for Java*)
    - JAXR (*Java API for XML Registries*)

Desenvolvimento

- Exemplo de Web Service em Java

```
@WebService()
public class Tempo {
   @WebMethod(name="getTemperaturaMinima")
   public float getTemperaturaMinima(
       (@WebParam="nomeLocalidade") String nomeLocalidade )
   { /* código do método */ }

   @WebMethod(name="getTemperaturaMaxima")
   public float getTemperaturaMaxima(
       (@WebParam="nomeLocalidade") String nomeLocalidade )
   { /* código do método */ }
}
```

Informações Adicionais

- REST (*REpresentational State Transfer*)
  - Alternativa mais leve para o SOAP
  - Recursos são identificados por uma URI e acessados através de mensagens HTTP
    - GET: obtém o estado do recurso
    - POST: modifica o estado do recurso
    - PUT: cria um recurso
    - DELETE: remove um recurso
  - Objetivos: simplicidade e melhor desempenho
  - Exemplos: GET /aluno?curso=123
    POST /aluno/101123001

Informações Adicionais

- JSON (*JavaScript Object Notation*)
  - Alternativa ao uso de XML
  - Há *parsers* em várias linguagens (não somente em *JavaScript*)
  - Comparação com XML
    - JSON é mais compacto (~30-40%)
    - *Parsing* de dados em JSON é mais leve
    - XML possui um conjunto amplo de padrões para definição do esquema de dados e de espaços de nomes, especificação semântica, criptografia, assinatura digital, etc.

Informações Adicionais

- JSON – Exemplo:

```
{ "previsaoDoTempo": "http://ufsc.br/previsao",
  "data": "01/07/2008",
  "unidade": "Celsius",
  "localidade": [
    { "id": "1",
      "nome": "Florianópolis",
      "temperaturaMinima": "15.2",
      "temperaturaMaxima": "19.8"  },
    ...
  ]
}
```

Informações Adicionais

- Especificações que adicionam recursos e funcionalidades aos Web Services
  - WS-Addressing: endereçamento e roteamento de mensagens na camada de aplicação
  - WS-BPEL/WS-CDL: orquestração/coreografia de processos de negócio
  - WS-Coordination/WS-Transaction: execução de transações distribuídas entre serviços
  - WS-ReliableMessaging: entrega confiável de mensagens SOAP a serviços Web
  - WS-Security: mecanismos para controle de acesso, integridade e confidencialidade
  - ...

Informações Adicionais

- Serviços Web Semânticos
  - Aperfeiçoam a descrição e descoberta de serviços usando tecnologias da Web Semântica
  - Propiciam a interação entre serviços sem intervenção humana
  - Dados e serviços são descritos usando ontologias, que representam um conjunto de conceitos dentro de um domínio de aplicação
  - Padrões relacionados:
    - OWL-S: *Ontology Web Language for Services*
    - WSMO: *Web Services Modeling Ontology*