WEB - Services

FELIPE CUNHA

Service Oriented Architecture – SOA

SOA é uma arquitetura que representa funcionalidades do software como serviços

Neste modelo de arquitetura os principais requisitos viram serviços e são acessados por outros serviços

Modularização e aumento da coesão dos componentes da aplicação

Já existiam tecnologias para SOA

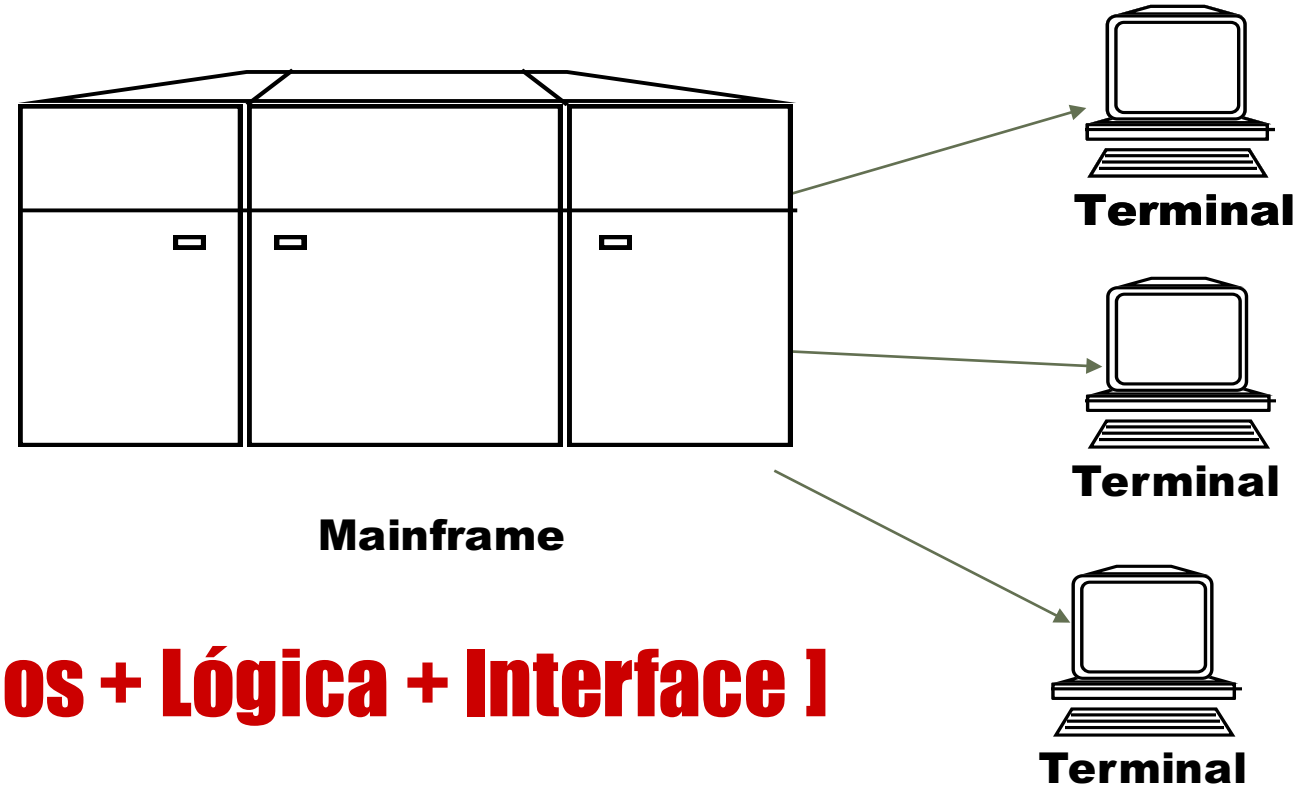
- Ex.: CORBA, RMI, Web Service, etc...

Interoperabilidade é muito importante

- Padronização
- Fraco acoplamento

Histórico das Arquiteturas

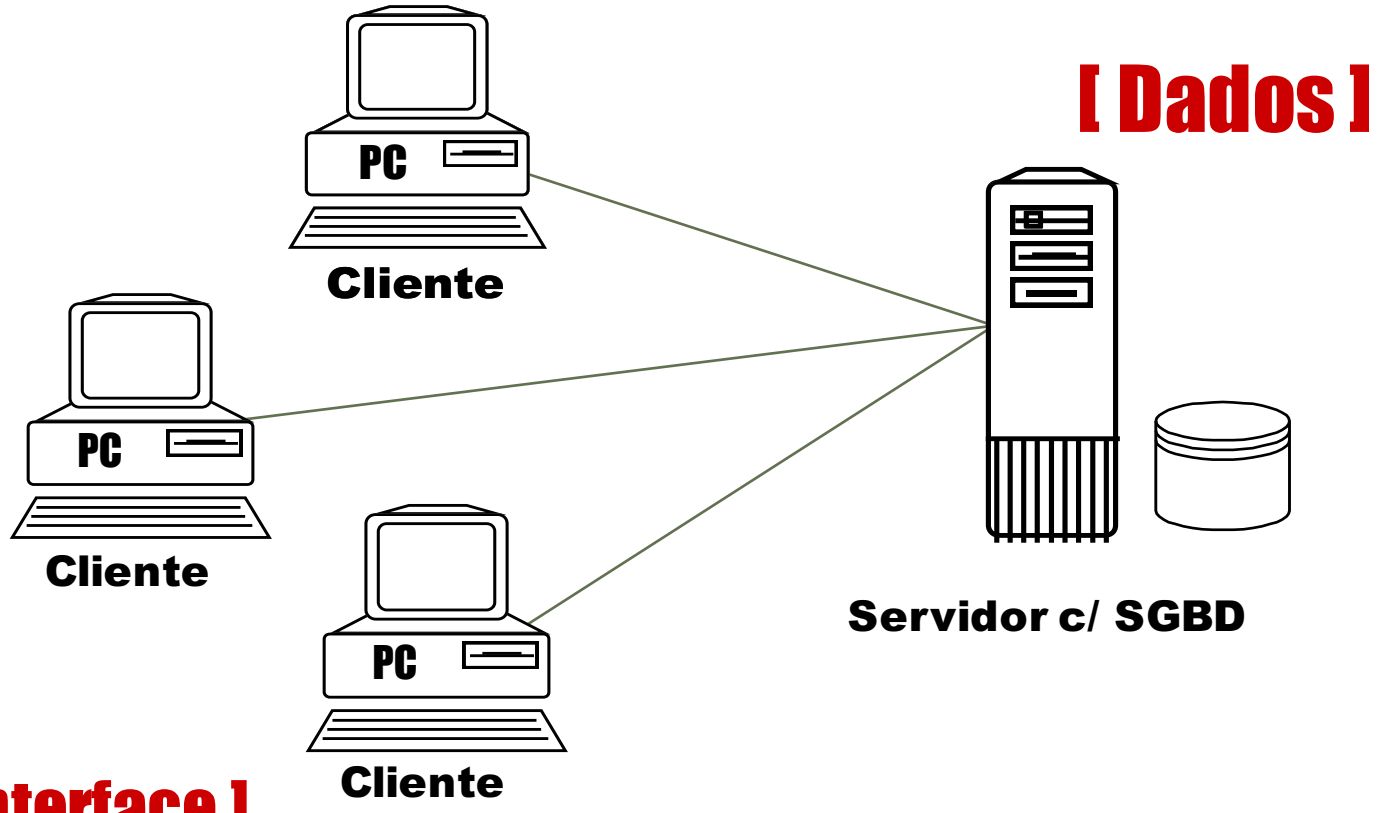
Mainframes



[Dados + Lógica + Interface]

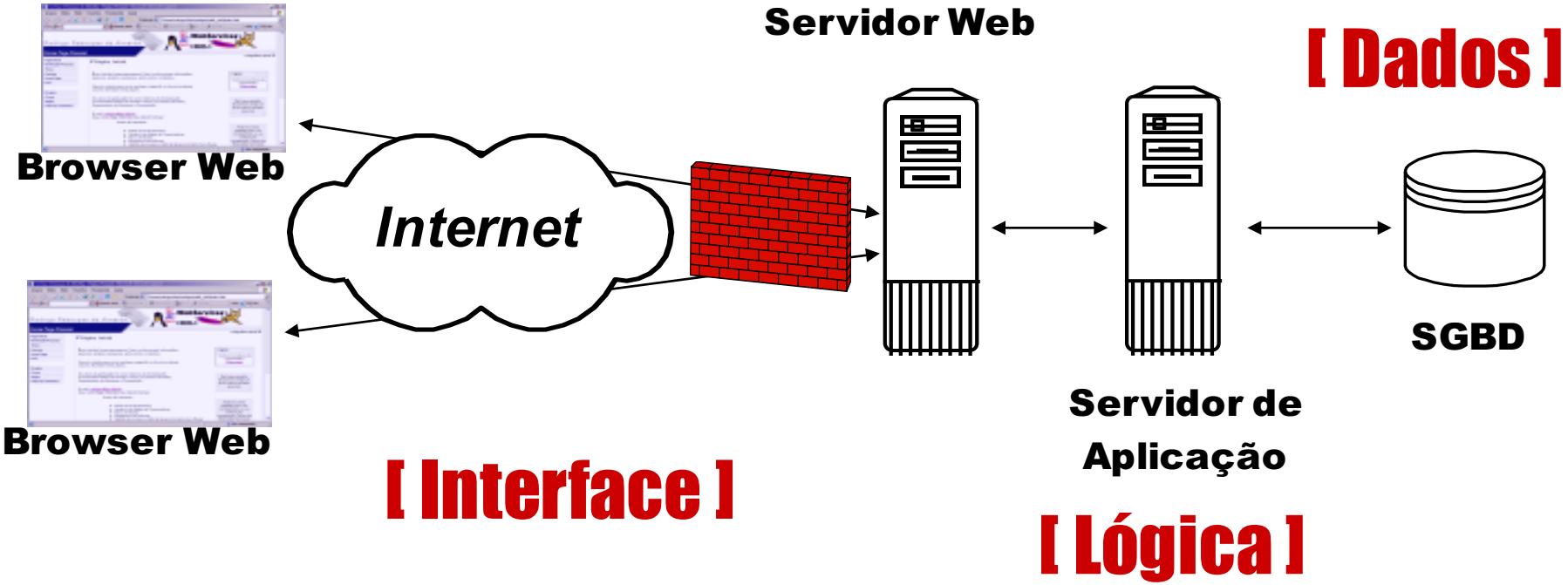
Histórico das Arquiteturas

Arquitetura Cliente-Servidor



Histórico das Arquiteturas

Arquitetura em N-Camadas



Arquitetura Orientada a Serviço

Modelo arquitetural para construção de aplicativos que promove:

- Um baixo acoplamento entre os componentes que podem ser reusados e trabalhos juntos como uma arquitetura distribuída

SOA – Principais Conceitos

Serviços: fornecem as funcionalidades do negócio

Interfaces auto descritivas: independente de plataforma, separada da implementação contendo a descrição das operações do serviço

Troca de mensagens: troca de dados nas operações definidas pelas interfaces deve ser independe de plataforma

Comunicação síncrona e assíncrona: troca de mensagens realizadas por SOA deve suportar chamadas síncronas e assíncronas

SOA – Principais Conceitos

Baixo acoplamento: descrição dos serviços pelo uso de interfaces e protocolas independentes de linguagem e plataforma

Registro de serviços: funcionam como uma lista telefônica para serviços

Composição de serviços: serviços individuais podem ser agrupados para formar um serviço mais elaborado

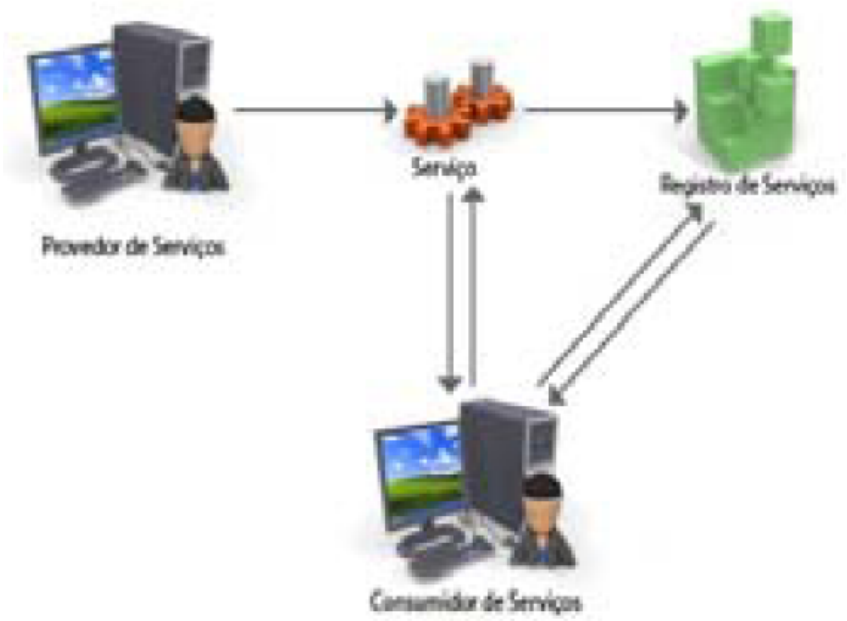
Qualidade de serviço (QoS): serviços podem possuir atributos de qualidade: confiabilidade, segurança, desempenho, etc

SOA – Papéis e Funções

Provedor
- descreve e publica seu serviço

Registro de Serviços
- mantém informações sobre serviços e suas localizações

Cliente
- localiza provedores de serviços através do registro de serviços



Computação Orientada a Serviço – SOC

SOC é um paradigma para a computação distribuída que define o modo com o qual são desenvolvidas, projetadas, disponibilizadas e "consumidas" aplicações

SOC é um paradigma computacional que utiliza serviços como unidades básicas para apoiar o desenvolvimento rápido, de baixo custo e de fácil composição de aplicações distribuídas até mesmo em ambientes heterogêneos

Computação Orientada a Serviço – SOC

Maior inovação é a mudança do paradigma

Orientado a Objetos

Orientado a Serviços

Paradigma Orientado a Objetos ⇔ ***Statefull***

×

Paradigma Orientado a Serviços ⇔ ***Stateless***

O Que é um Serviço ?

Serviço é uma entidade capaz de prover alguma capacidade para seus clientes através da troca de mensagens

Representa a unidade atômica do SOA

O Que é um Serviço ?

Serviço Windows

- Servidor DHCP, Serviço de Terminal, *Log* de Eventos, ...

Serviço de Software

- Serviços de *Middleware*
- Serviços Distribuídos
- RMI

Serviço de Negócio

- Serviço de Mapas: *Google Maps*
- *Flickr*

Serviço Web

Serviço Web (*Web Services*) é uma tecnologia de chamada remota de objetos

Fornece infraestrutura para desenvolvimento de aplicações distribuídas (Web ou não)

Permitem a criação de pequenos módulos de código reutilizáveis e disponibilizados para construção de aplicações "tipo-*LEGO*"

Utiliza protocolos Web como meio de transporte e comunicação

Alto grau de abstração em relação a linguagens de programação e plataformas de hardware / software

Serviço Web

Realização de um serviço na Web

Forma de acessar serviços descritos em XML por meio de WSDL

Tecnologia para construir sistemas distribuídos

Serviço Web

Distribuídos

Síncronos ou assíncronos

Suportam RPC

Acoplamento fraco

Auto descritivos

Suportam troca de documentos

Serviço Web

Um Serviço Web é um ponto de acesso a funcionalidade que pode ser:

- LOCALIZADO dinamicamente
- Ter sua interface DESCOBERTA automaticamente, porque o serviço sabe se descrever
- Ser CHAMADO na Web

Não faz parte do conceito de Serviço Web a criação de interfaces gráficas para os usuários

Serviço Web é a tecnologia ideal para comunicação entre sistemas (aplicações B2B)

Independência de plataforma e linguagem de programação

XML é a base de tudo

Oferece um formato ASCII para trocar qualquer tipo de informação estruturada

Usa o "estilo" HTML de *markup* com *tags*

- <pessoa nome="João">
  <frutasFavoritas>
  <fruta>Manga</fruta>
  <fruta>Maçã</fruta>
  <fruta>Uva</fruta>
  </frutasFavoritas>
  </pessoa>

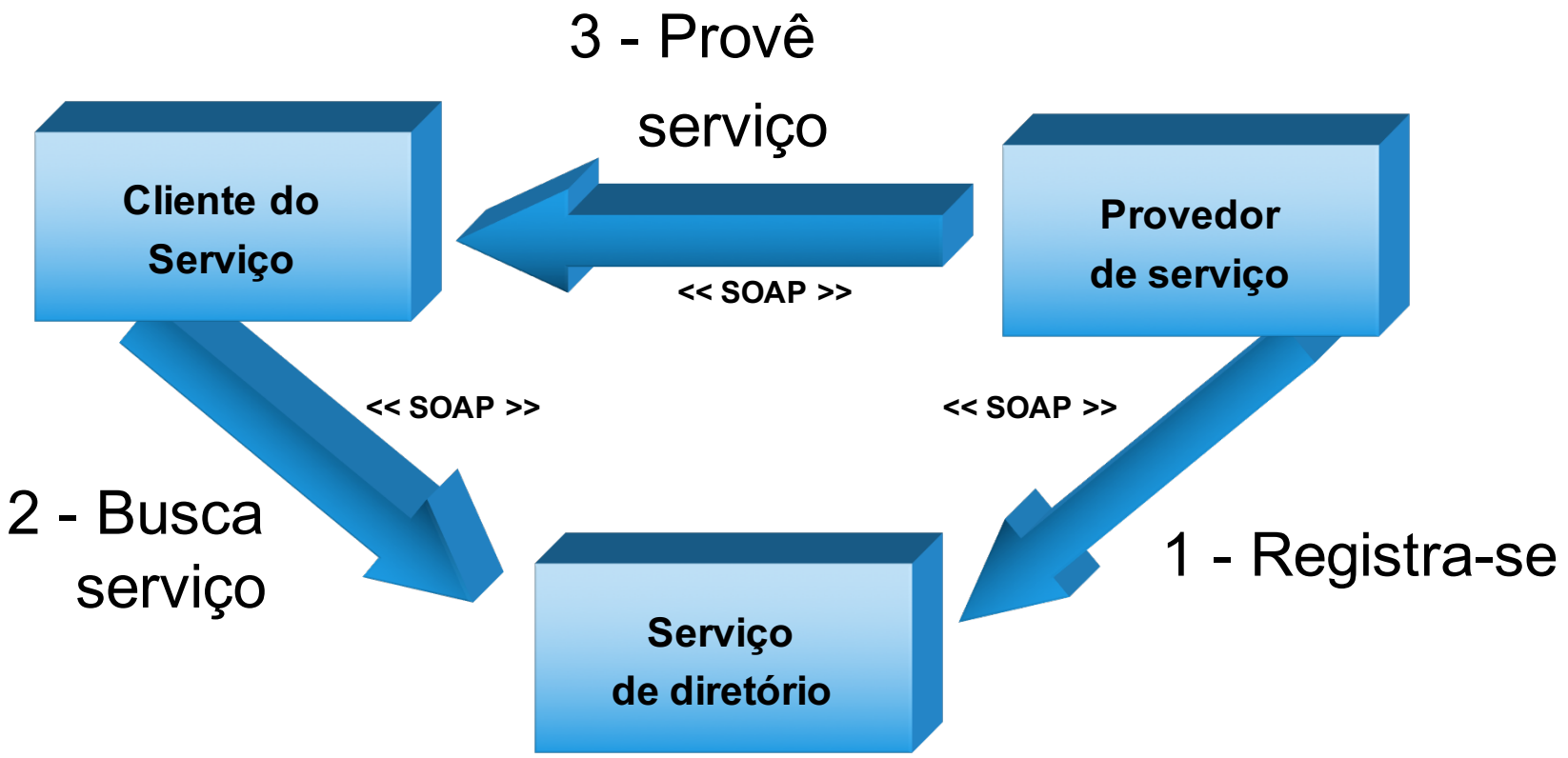
Um Serviço Web será descrito e publicado em um arquivo XML de acordo com WSDL

Serviço Web

Se fundamenta basicamente em três tecnologias

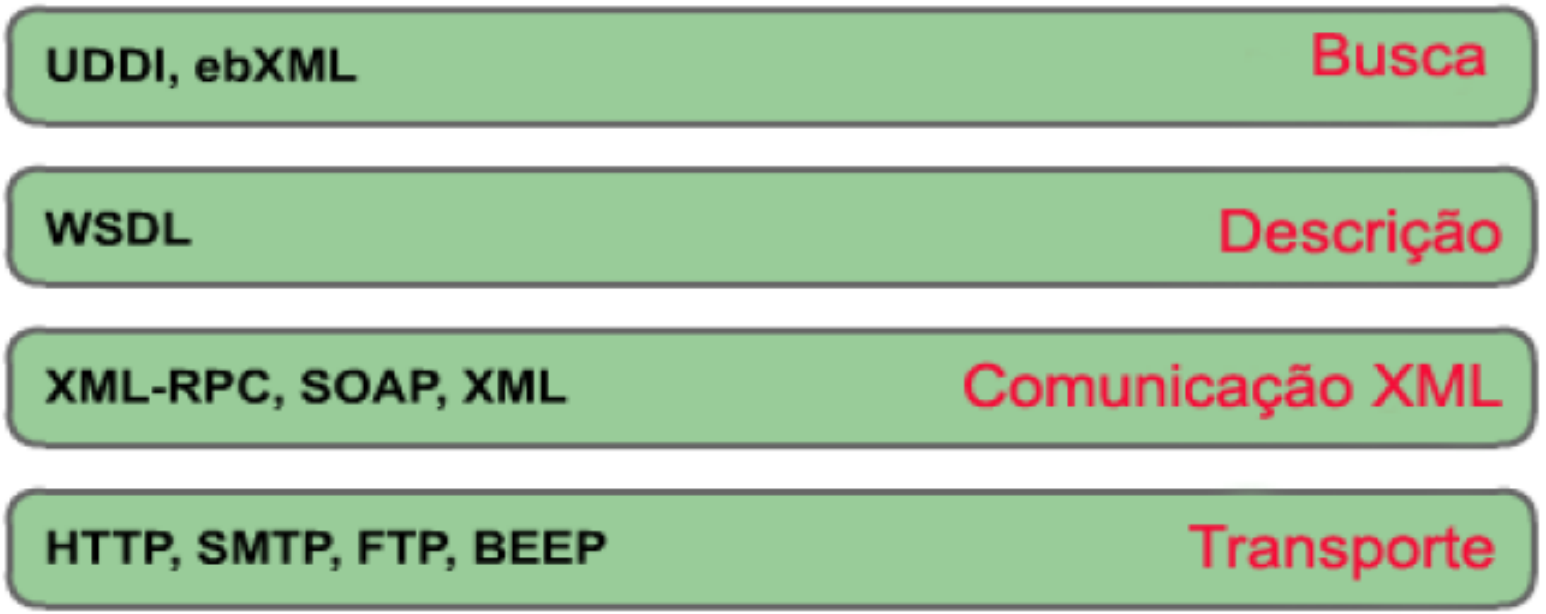
- Simple Object Access Protocol (SOAP)
  - Um protocolo baseado em XML que permite que os clientes se comuniquem com os provedores de serviço
- Web Services Description Language (WSDL)
  - Linguagem para definição/descrição da interface de acesso ao serviço
- Universal Description, Discovery and Integration (UDDI)
  - Permite o registro dos Serviços Web possibilitando que outras aplicações os encontrem

Serviços Web – Arquitetura



Pilha de Protocolos de um Serviço Web

Pode-se dividir os protocolos associados a tecnologia de Serviços Web em 4 camadas



Camada de Comunicação – SOAP

Serviços Web são identificados por uma URI (Unique Resource Identifier) e são descritos e definidos usando XML

SOAP (Simple Object Access Protocol) padrão para a troca de mensagens entre aplicações e Serviços Web, já que é uma tecnologia construída com base em XML e HTTP

SOAP é um procotolo projetado para invocar aplicações remotas através de RPC

SOAP é, portanto, um padrão normalmente aceito para utilizar-se com Serviços Web

Características do SOAP

Definido pelo consórcio W3C

Protocolo baseado em XML para a troca de informações em um ambiente distribuído

Padrão de utilização com Serviços Web

Normalmente utiliza HTTP e SMTP como protocolo de transporte

É mais utilizado sobre HTTP pois consegue atravessar firewalls

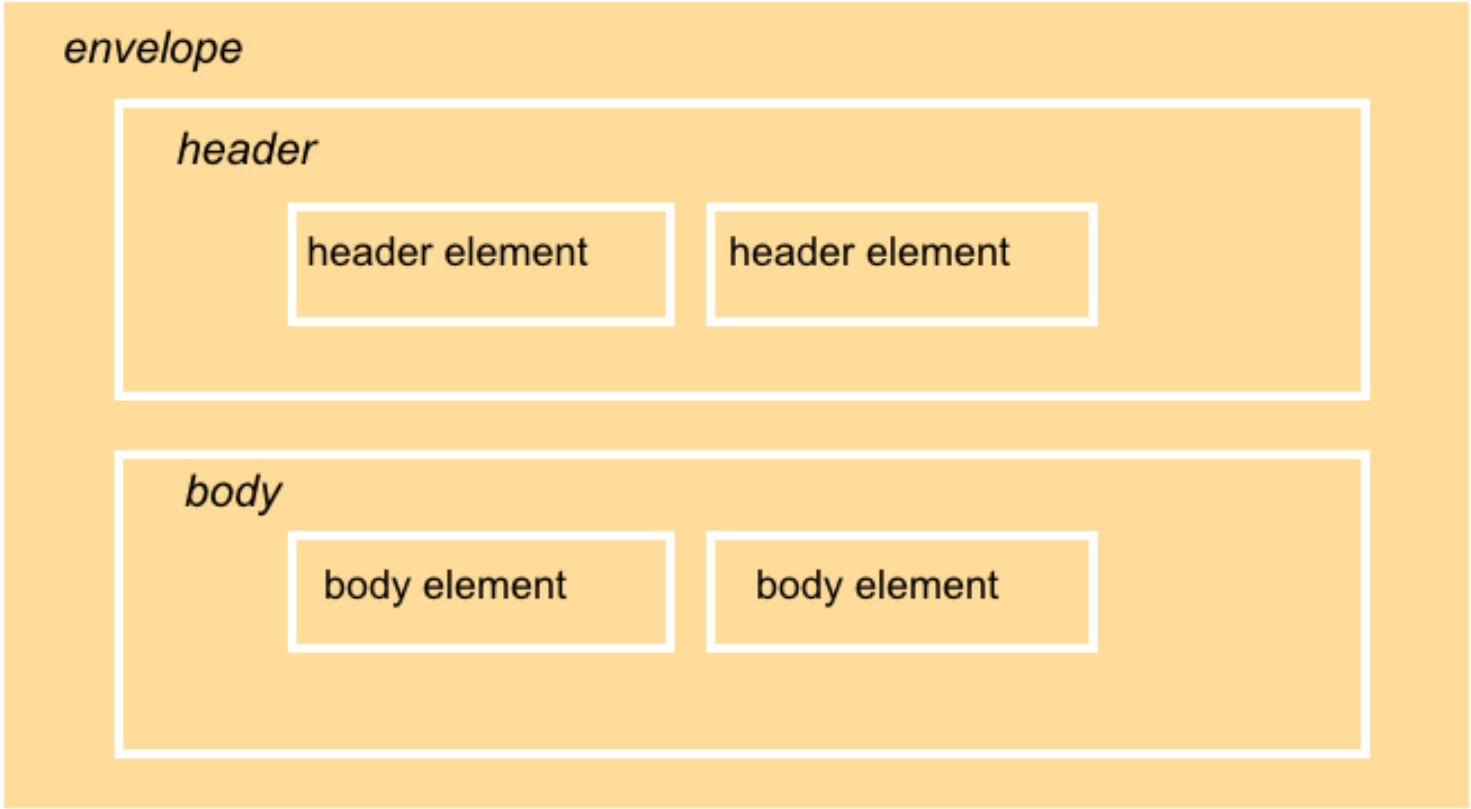
Estrutura da Mensagem SOAP

Envelope: toda mensagem SOAP deve contê-lo pois ele representa o elemento raiz do documento XML

Header: é um cabeçalho opcional que carrega informações adicionais como, por exemplo, se a mensagem deve ou não ser processada por um determinado nó intermediário (se utilizado, o Header deve ser o primeiro elemento do Envelope)

Body: este elemento é obrigatório e contém o *payload* ou a informação a ser transportada para o seu destino final

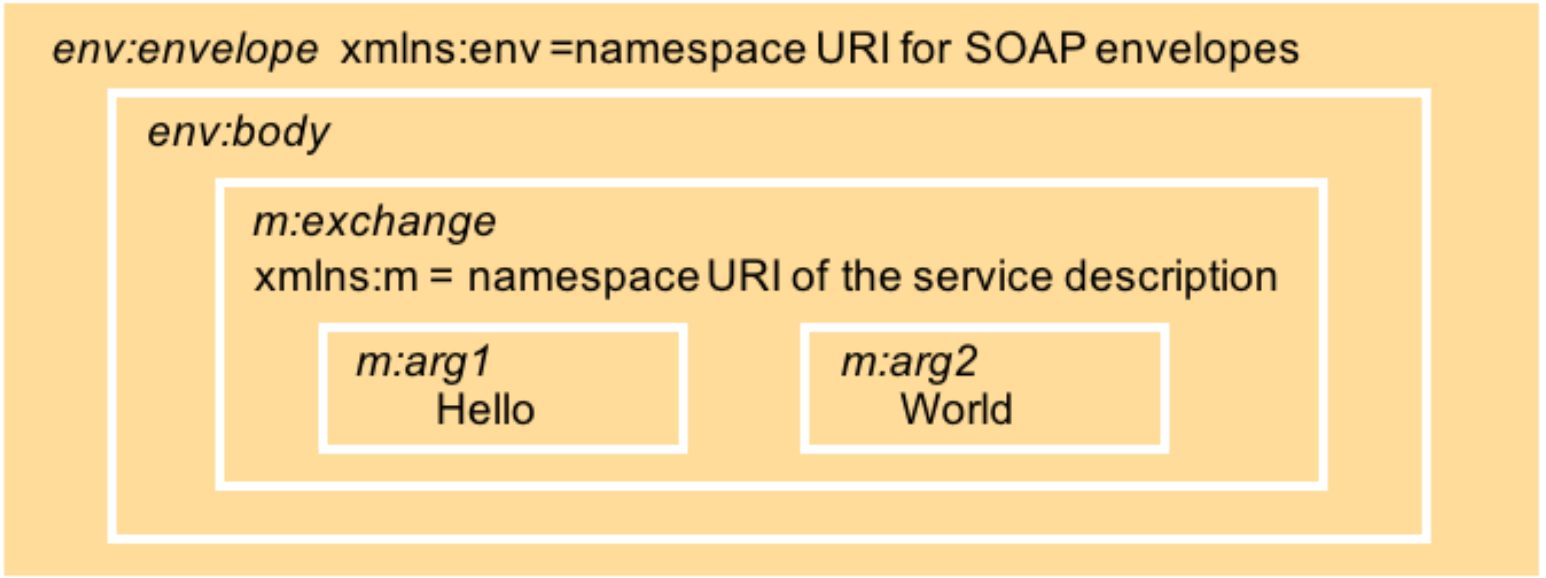
Mensagem SOAP em um envelope



Estrutura de Mensagem SOAP

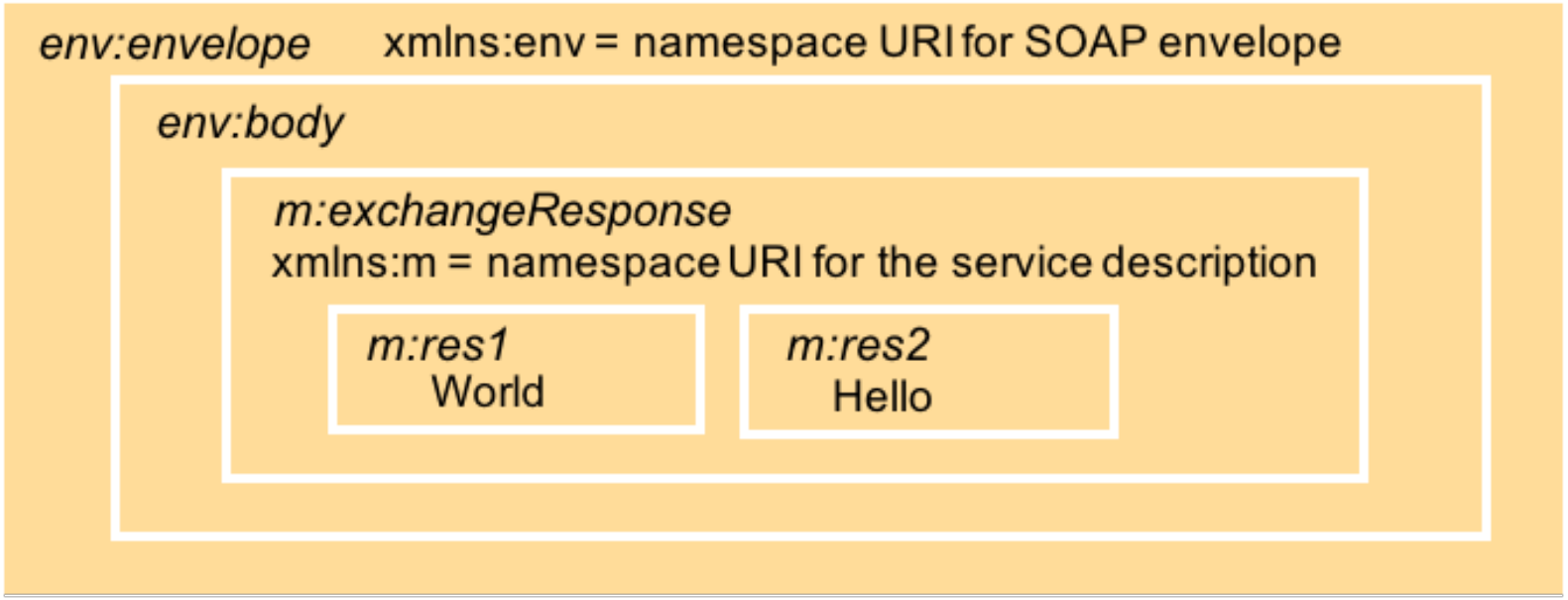
```
<SOAP:Envelope xmlns:SOAP=
    http://schemas.xmlsoap.org/soap/envelope/>

   <SOAP:Header>
      <!conteudo do cabecalho >
   </SOAP:Header>

   <SOAP:Body>
      <!conteudo do corpo>
   </SOAP:Body>

</SOAP:Envelope>
```

Exemplo de uma requisição simples sem os cabeçalhos

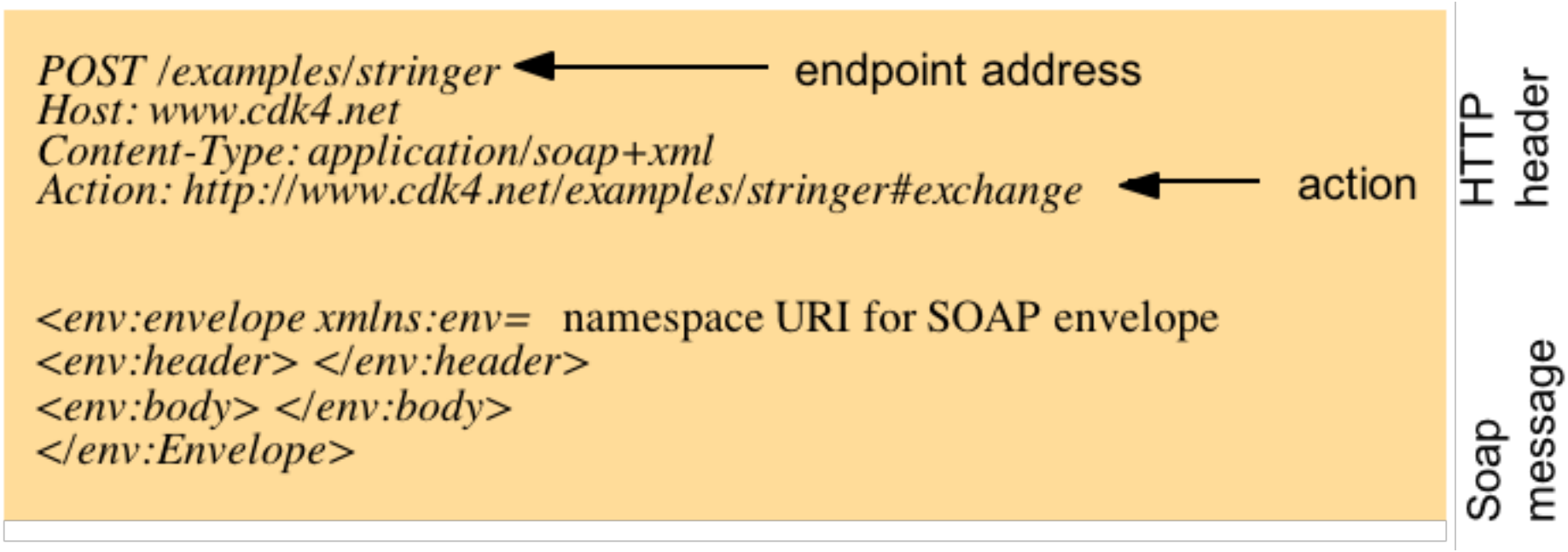


Cada elemento XML é representado por um bloco com seu nome seguido pelos seus argumentos e conteúdo

Exemplo de uma resposta (reply) para uma requisição como a anterior



Uso de uma requisição HTTP POST na comunicação cliente-servidor do SOAP



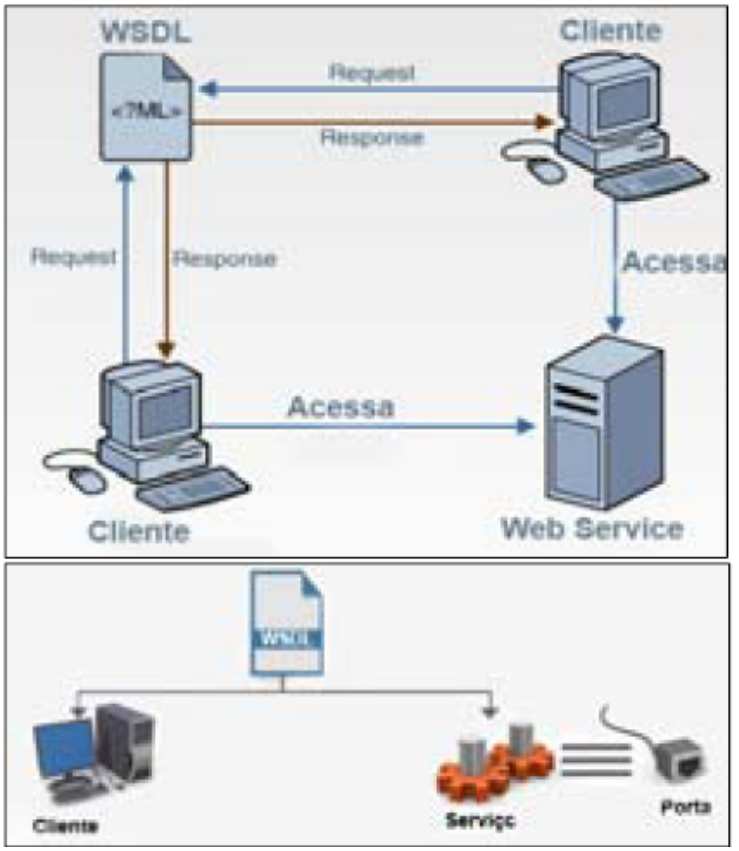
Camada de Descrição – WSDL

WSDL é um documento XML que fornece informações sobre o Serviço Web de maneira independente de linguagem e plataforma

Clientes precisam saber como acessar um Serviço Web

- Qual a operação
- Quais os parâmetros
- Qual o endereço

WSDL define serviços por meio de portas em que cada porta está associada a um serviço específico



Documento WSDL

WSDL ≡ Web Services Description Language

Pronunciado “wisdle” = uisdal

É uma linguagem XML que contém informação sobre a interface, a semântica, e outros detalhes de chamadas a um Serviço Web

Em resumo: Linguagem XML para descrever um Serviço Web

Documento WSDL

Um documento WSDL define um XML Schema para descrever um Serviço Web

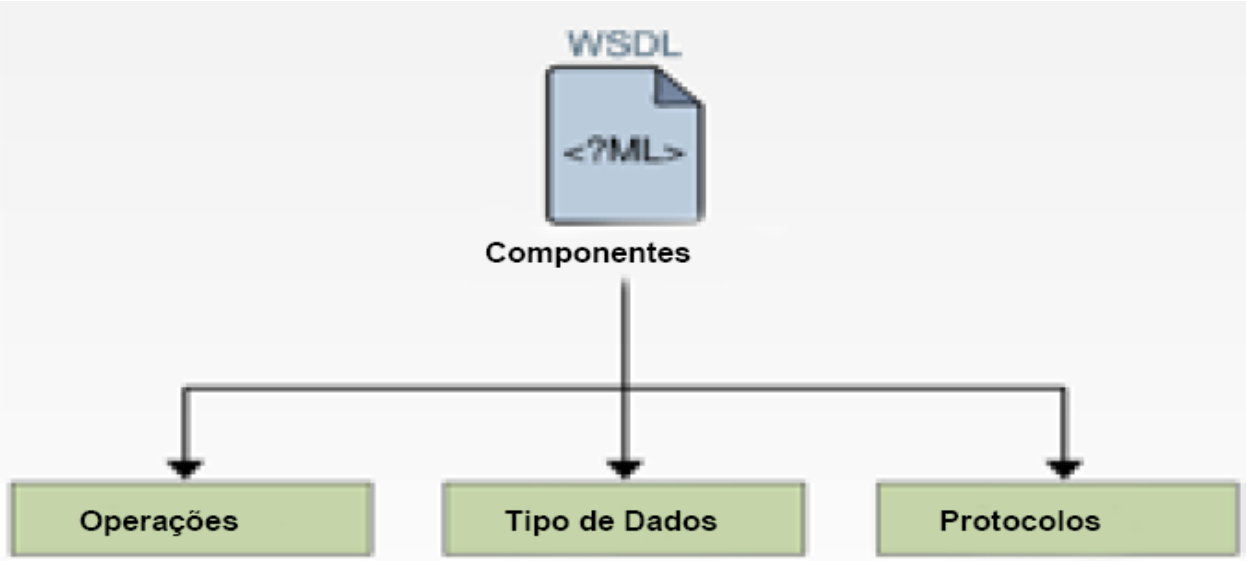
Cliente enviando uma mensagem a um Serviço Web:

- Obtém a descrição do serviço (WSDL)
- Constrói a mensagem, passando os parâmetros corretos baseados no documento
- Mensagem enviada para o endereço onde o serviço está localizado
- O Serviço Web quando recebe a mensagem, valida a mesma baseado no WSDL
- Executa o serviço e responde ao cliente

WSDL – Especificação

Um documento WSDL é formado por componentes

- Operações
- Tipo de Dados (*XML Schema*)
- Protocolos

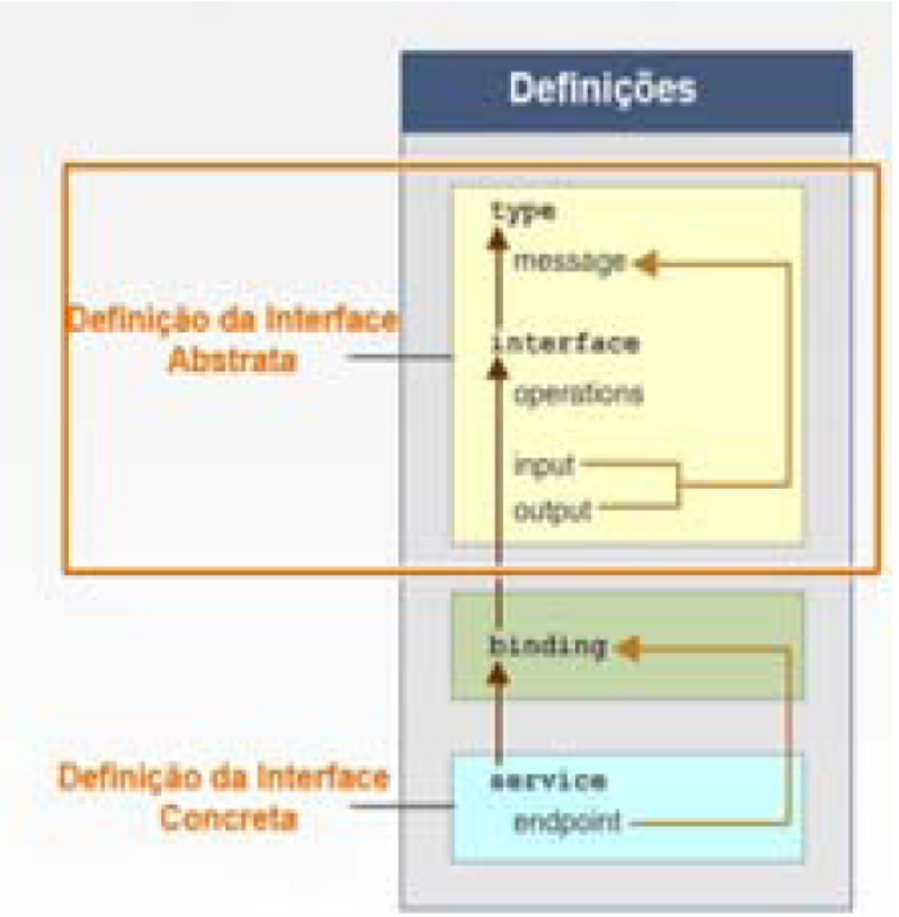


Estrutura de um Documento WSDL

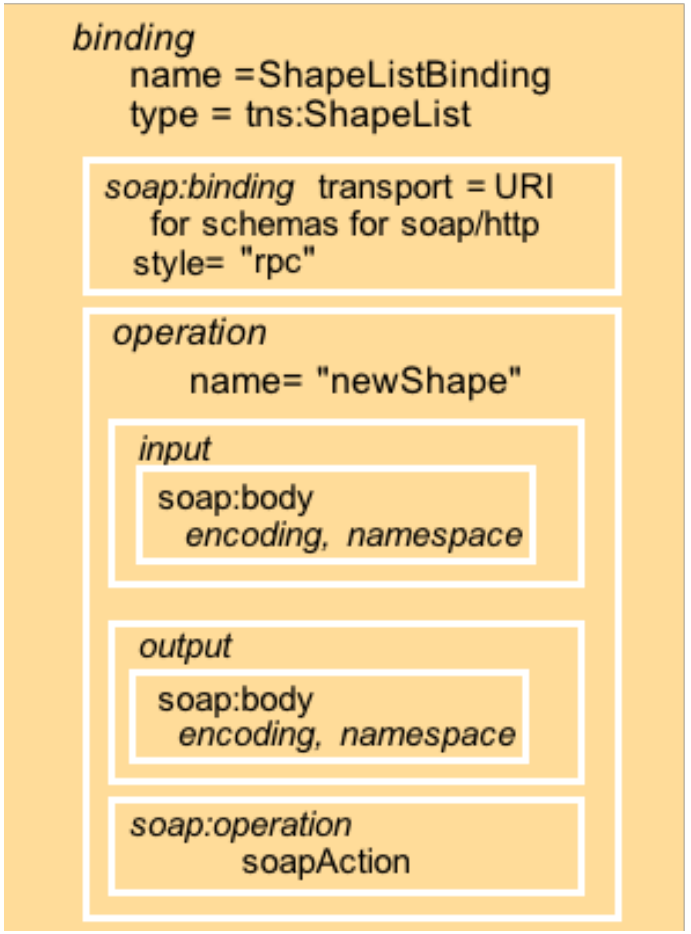
Elemento raiz:
- <definitions>

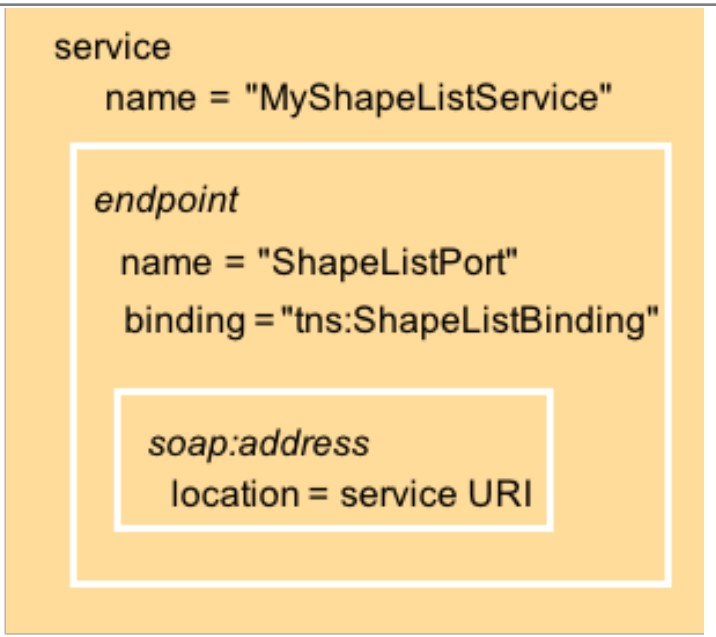
A interface abstrata fornece uma descrição genérica da interface do Serviço Web
- tipo dos dados
- formato de mensagens
- operações

A interface concreta fornece detalhes sobre a implementação do Serviço Web
- protocolo usado para transporte de mensagens
- endereço IP onde o Serviço Web está localizado



Definições do serviço e do binding SOAP





the service URI is:
"http://localhost:8080/ShapeList-jaxrpc/ShapeList"

Exemplo WSDL – Definição de Tipos

```
<?xml version='1.0' encoding='UTF-8'?>
<definitions xmlns:wsu="http://docs.oasis-
open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd"
xmlns:wsp="http://www.w3.org/ns/ws-policy"
xmlns:wsp1_2="http://schemas.xmlsoap.org/ws/2004/09/policy"
xmlns:wsam="http://www.w3.org/2007/05/addressing/metadata"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:tns="http://webservice.teste.my/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns="http://schemas.xmlsoap.org/wsdl/"
targetNamespace="http://webservice.teste.my/" name="TesteService">
<types>
<xsd:schema>
<xsd:import namespace="http://webservice.teste.my/"
schemaLocation="http://localhost:8080/TesteInicial/TesteService?xsd=1" />
</xsd:schema>
</types> ...
```

Exemplo WSDL – *XML Schema*

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

<xs:element name="TestOp" type="tns:TestOp" />
<xs:element name="TestOpResponse" type="tns:TestOpResponse" />

<xs:complexType name="TestOp">
   <xs:sequence>
      <xs:element name="param1" type="xs:string" minOccurs="0" />
   </xs:sequence>
</xs:complexType>
<xs:complexType name="TestOpResponse">
   <xs:sequence>
      <xs:element name="return" type="xs:string" minOccurs="0" />
   </xs:sequence>
</xs:complexType>
</xs:schema>
```

Exemplo WSDL – Definição de Msgs

```
...
<message name="TestOp">
   <part name="parameters" element="tns:TestOp" />
</message>
<message name="TestOpResponse">
   <part name="parameters" element="tns:TestOpResponse" />
</message>

<portType name="TestWebService">
   <operation name="TestOp">
       <input
       wsam:Action="http://webservice.teste.my/TestWebService/
                        TestOpRequest" message="tns:TestOp" />
              <output
       wsam:Action="http://webservice.teste.my/TestWebService/
                        TestOpResponse" message="tns:TestOpResponse" />
       </operation>
</portType>
...
```

Exemplo WSDL – Definição do Serviço

```
...
<binding name="TestWebServicePortBinding" type="tns:TestWebService">
  <soap:binding transport="http://schemas.xmlsoap.org/soap/http"
        style="document" />
   <operation name="TestOp">
      <soap:operation soapAction="" />
      <input> <soap:body use="literal" /> </input>
      <output> <soap:body use="literal" /> </output>
   </operation>
</binding>
<service name="TesteService">
   <port name="TestWebServicePort"
                    binding="tns:TestWebServicePortBinding">
   <soap:address
location="http://localhost:8080/TesteInicial/TesteService" />
   </port>
</service>
</definitions>
```

Camada de Busca – UDDI

Framework independente de plataforma usado na comunicação entre provedores de serviço e consumidores

UDDI descreve como criar registro para armazenar as informações sobre Serviço Web

Funciona como uma lista telefônica de Serviços Web

UDDI (Localizar Serviços)

UDDI ≡ Universal Description, Discovery, and Integration

Permite cadastrar serviços e localizá-los

Não é necessário usar UDDI se o cliente já tiver o documento WSDL

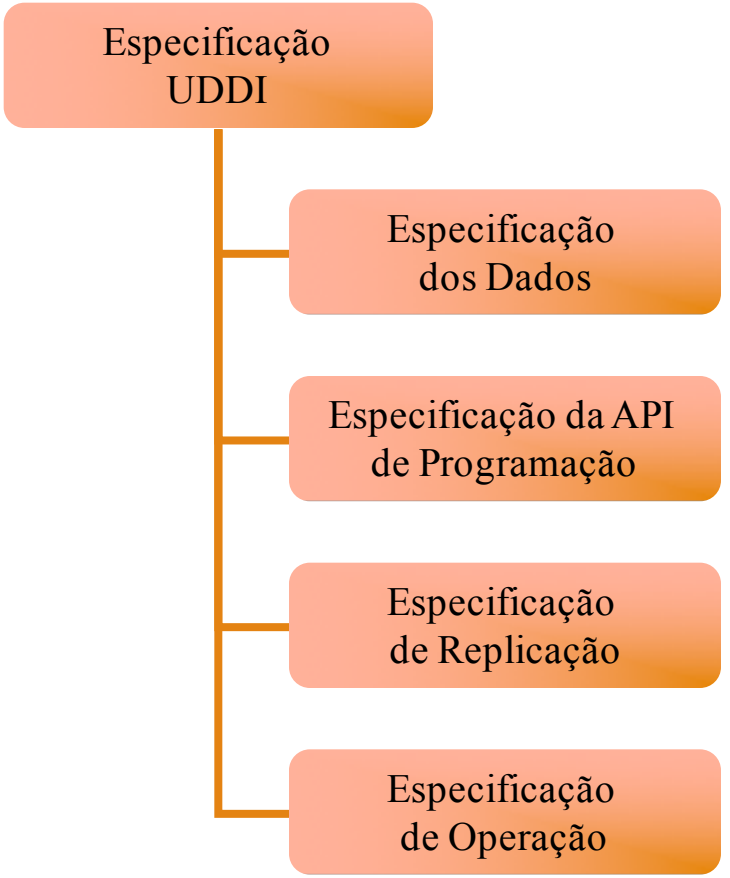
UDDI – Especificação

A especificação dos dados descreve como as informações são armazenadas no registro

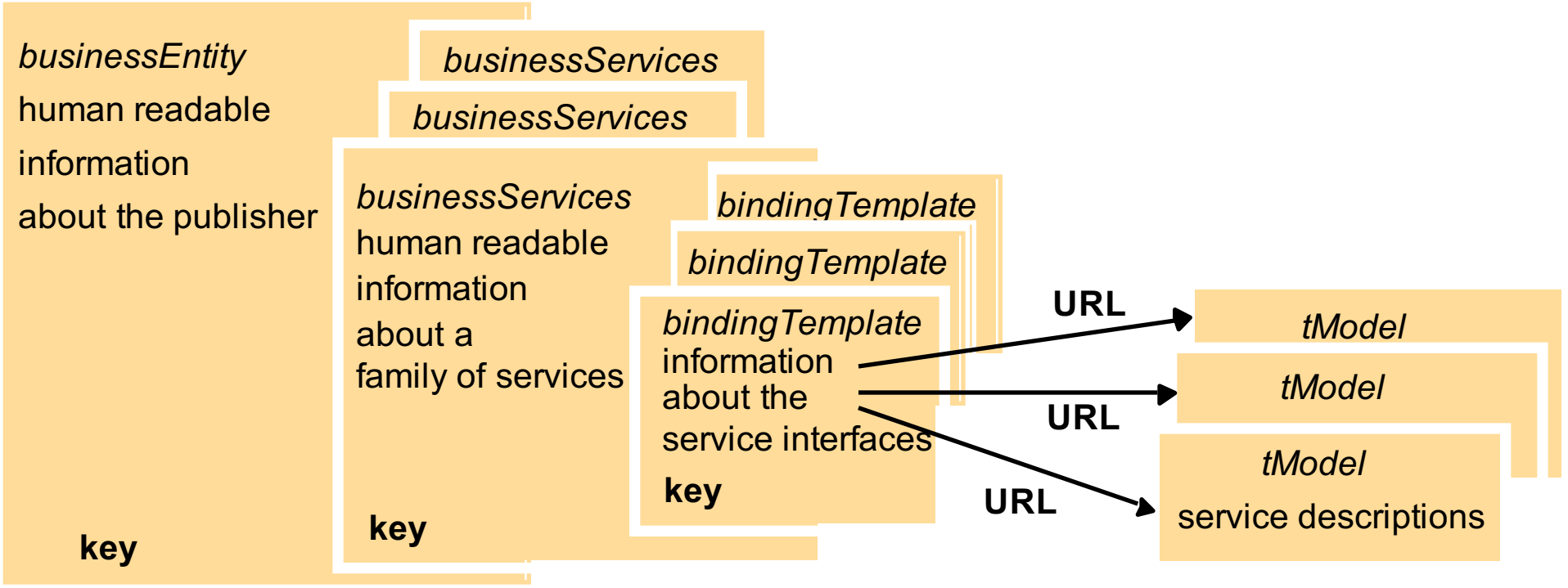
A especificação da API descreve como um registro UDDI pode ser acessado através das APIs de publicação e consulta

A especificação para replicação descreve como as informações dos registros são replicadas entre eles.

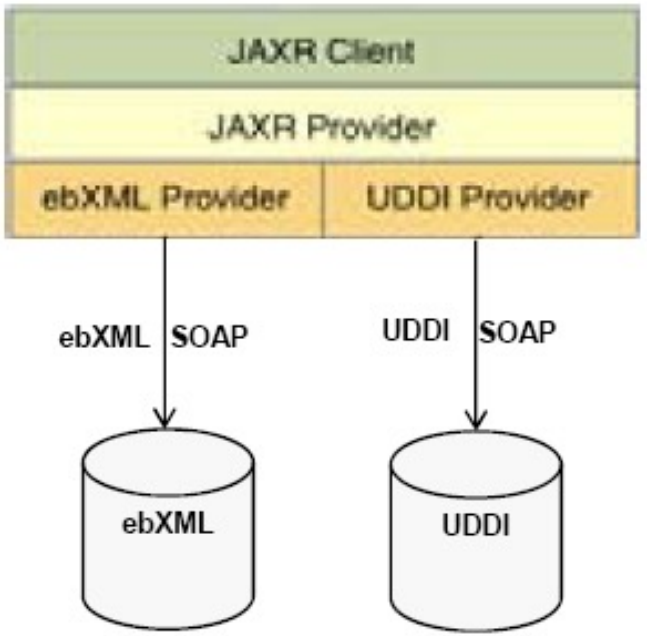
A especificação de operação define políticas para gerenciamento e segurança.



Estruturas de dados principais do UDDI



Java API para UDDI



JAXR (*Java API for XML Registries*) fornece um modo padrão para acessar diferentes tipos de repositórios UDDI

JAXR é dividida em

- JAXR *Client*
- JAXR *Provider*
- *Registry Provider*
- *Registries*

Ferramentas para Desenvolvimento de Serviços Web

Ferramentas da Oracle/Sun

- JWSDP (Java WS Developer Pack) – várias APIs
  - JAXP, JAXB, JAX-RPC, SAAJ, JAXR, JWSDL

JBossWS

XFire

Ferramentas da Apache

- Axis do projeto Apache
  - WSDL2Java converte WSDL para Java

Java web service interface ShapeList

```
import java.rmi.*;
public interface ShapeList extends Remote {
        int newShape(GraphicalObject g) throws RemoteException; 1
        int numberOfShapes()throws RemoteException;
        int getVersion() throws RemoteException;
        int getGOVersion(int i)throws RemoteException;
        GraphicalObject getAllState(int i) throws RemoteException;
}
```

Java implementation of the ShapeList server

```
import java.util.Vector;
public class ShapeListImpl implements ShapeList {
        private Vector theList = new Vector();
        private int version = 0;
        private Vector theVersions = new Vector();
        public int newShape(GraphicalObject g) throws RemoteException{
                version++;
                theList.addElement(g);
                theVersions.addElement(new Integer(version));
                return theList.size();
        }
        public int numberOfShapes(){}
        public int getVersion() {}
        public int getGOVersion(int i){ }
        public GraphicalObject getAllState(int i) {}
}
```

Java implementation of the *ShapeList* client

```
package staticstub;
import javax.xml.rpc.Stub;

public class ShapeListClient {
        public static void main(String[] args) { /* pass URL of service */
          try {
                Stub proxy = createProxy();                                    1
                proxy._setProperty                                             2
                  (javax.xml.rpc.Stub.ENDPOINT_ADDRESS_PROPERTY, args[0]);
                ShapeList aShapeList = (ShapeList)proxy;                       3
                GraphicalObject g = aShapeList.getAllState(0);                 4
          } catch (Exception ex) { ex.printStackTrace(); }
        }
        private static Stub createProxy() {                                    5
                return
                  (Stub) (new MyShapeListService_Impl().getShapeListPort());   6
   }
}
```

WSDL request and reply messages for the newShape operation

message name = "ShapeList_newShape"

part name ="GraphicalObject_1"
type = "ns:GraphicalObject "

message name = "ShapeList_newShapeResponse"

part name= "result"
type = "xsd:int"

tns – target namespace

xsd – XML schema definitions

WSDL da operação newShape

operation name = "newShape"
pattern = In-Out

input message = tns:ShapeList_newShape

output message ="tns:ShapeList_newShapeResponse"

tns – target namespacexsd – XML schema definitions

Padrões de troca de mensagens para operações WSDL

Nome	*Mensagens enviadas por*			
	Client	*Server*	*Entrega*	*Mensagem de falha*
In-Out	*Request*	*Reply*		may replace *Reply*
In-Only	*Request*			no fault message
Robust In-Only	*Request*		guaranteed	may be sent
Out-In	*Reply*	*Request*		may replace *Reply*
Out-Only		*Request*		no fault message
Robust Out-Only		*Request*	guaranteed	may send fault

Exemplo: Center for Operational Oceanographic Products and Services / NOAA

Dados oceanográficos:

nível da água, dados sobre correntes, etc.

http://opendap.co-ops.nos.noaa.gov/axis/

Center for Operational Oceanographic Products and Services

CO-OPS SOAP Web Services

Take the Survey

This is a listing of on-line data that is accessible through Web Services and it is provided by the Center for Operational Oceanographic Products and Services.

For a listing of on-line data that is accessible using the IOOS SOS Web Services provided by the Center for Operational Oceanographic Products and Services **click here**

Water Level					
Preliminary Data					
Six Minute Data	WSDL	SOAP Request	SOAP Response	JAVA Client	Try me
One Minute Data	WSDL	SOAP Request	SOAP Response	JAVA Client	Try me
Verified Data					
Six Minute Data	WSDL	SOAP Request	SOAP Response	JAVA Client	Try me
Hourly Height Data	WSDL	SOAP Request	SOAP Response	JAVA Client	Try me

REST

Representational State Transfer

Estilo arquitetural (nao um padrão)

Ideia: uma rede de páginas web onde o cliente progride atraves da aplicação de links selecionados

Utiliza o padrão HTTP

Stateless: cada requisição deve conter toda a informação necessária

Interface uniforme: todos os recursos são acessados com uma interface genérica baseada em HTTP

Componentes em camada: intermediários como proxies, caches, etc melhoram a performance e a segurança

SOAP vs. REST

REST – clientes submetem requisições para Web Services como requisições HTTP

SOAP – clientes submetem requisições na forma de um documento XML

Vantagens do SOAP:

Suporta uma variedade de protocolos (transporte, autenticação, criptografia, etc)

Vantagens do REST:

- Simples, baseado na pilha HTTP
- Muitos sites suportam ambos os padrões