UNIVERSIDADE FEDERAL DO CEARÁ - Campus Quixadá
Cursos: SI, ES, RC, CC e EC
Código: QXD0043
Disciplina: Sistemas Distribuídos

Capítulo 9 – Serviços Web

Prof. Rafael Braga

Introdução

- A *World Wide Web* teve como objetivo inicial permitir a troca de documentos entre os computadores distribuídos por essa rede.
- Porém, com o crescimento e popularização desta, passou-se a utilizá-la como base para comunicação entre aplicações distribuídas que necessitam de um método eficiente para intercâmbio de dados (informações).

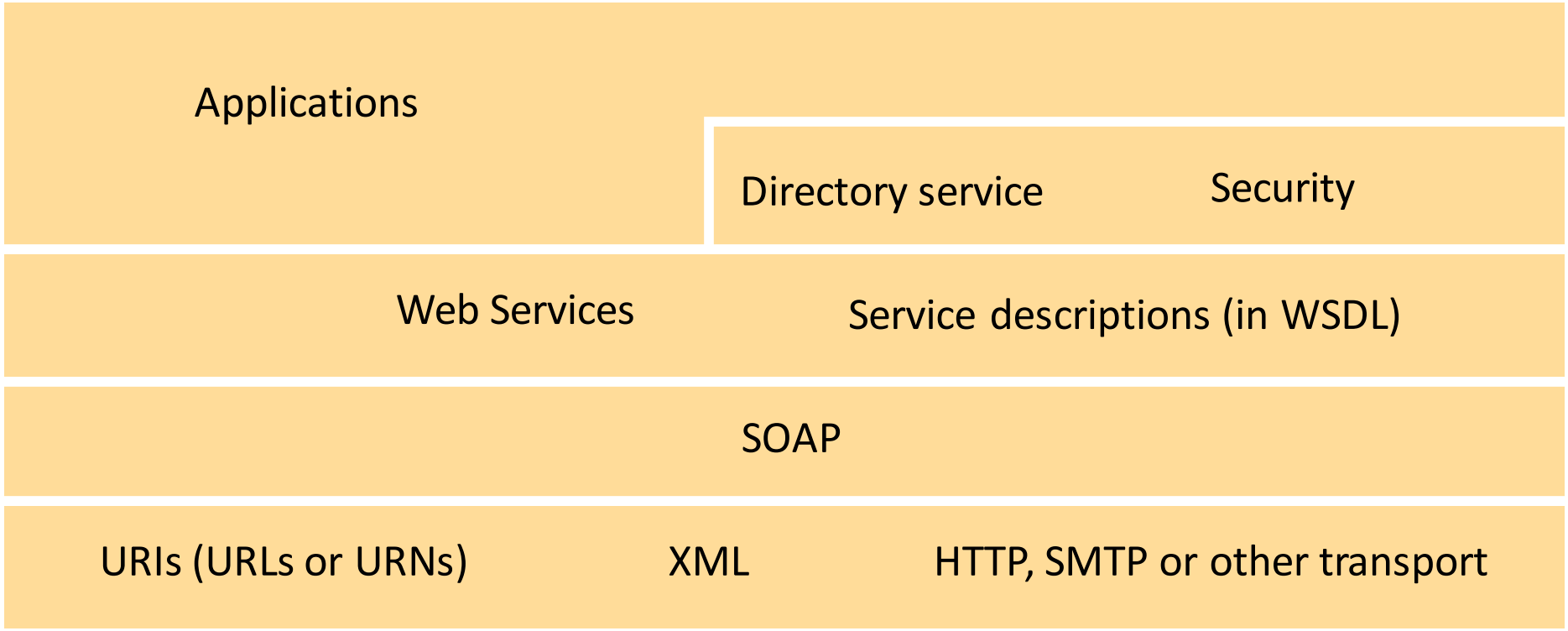
Introdução

- Consequentemente, uma parcela significativa dos sistemas computacionais passou a utilizar este modelo de comunicação, permitindo vislumbrar a integração destes, uma vez que uma das maiores dificuldades existentes atualmente no mundo computacional é a **integração de sistemas**.
  - Servidor web x serviço web;
- A representação de dados externa e o empacotamento da mensagens trocadas são feitos em XML.
  - SOAP

Introdução

- A arquitetura dos serviços web (*web services*) busca a solução sobre o problema da integração, pois ao utilizar **padrões abertos** de protocolos e linguagens, possibilita a integração das mais diversas aplicações distribuídas sem se preocupar com a heterogeneidade intrínseca dos ambientes distribuídos (como a Web, por exemplo).
  - URI (URL's ou URN's), HTTP, TCP, SMTP

Infraestrutura e componentes dos serviços web



Introdução

- Um serviço web fornece uma *descrição do serviço*. Nesse caso, especificamente, o WSDL (*Web Services Description Language*).
- Serviço de atribuição de nomes ou de diretórios para descoberta dos serviços. Páginas amarelas..
  - Uma busca no google
- Segurança da XML com documentos cifrados ou assinados..

Introdução

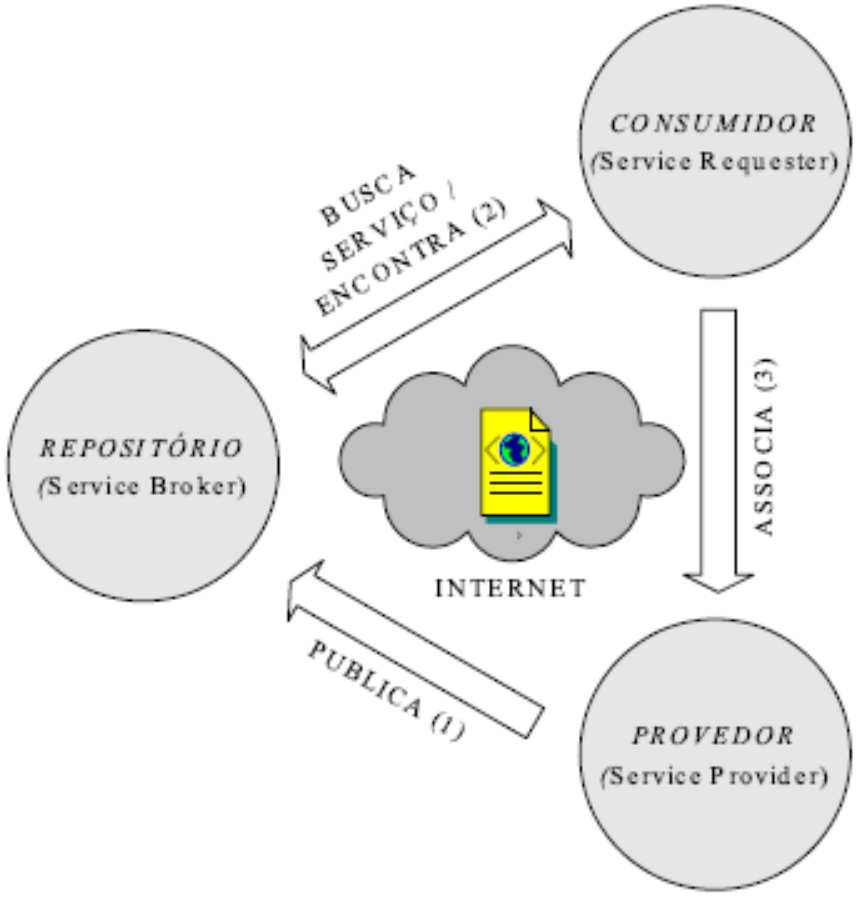
- Além disso, Web Services são um meio capaz de prover a interação de aplicações e programas sem a intermediação do homem, isto é, web services podem se comunicar com outros web services, podendo assim fornecer novas aplicações com os serviços provenientes destes.

Serviços Web

- Composto por um conjunto de operações;
  - Programas, objetos, banco de dados..
- A principal característica é que podem processar mensagens SOAP formatadas em XML.
  - Uma alternativa é o REST. Exemplos, os serviços da Amazon, eBay (leilões ou *sniping*) e etc.

Serviços web

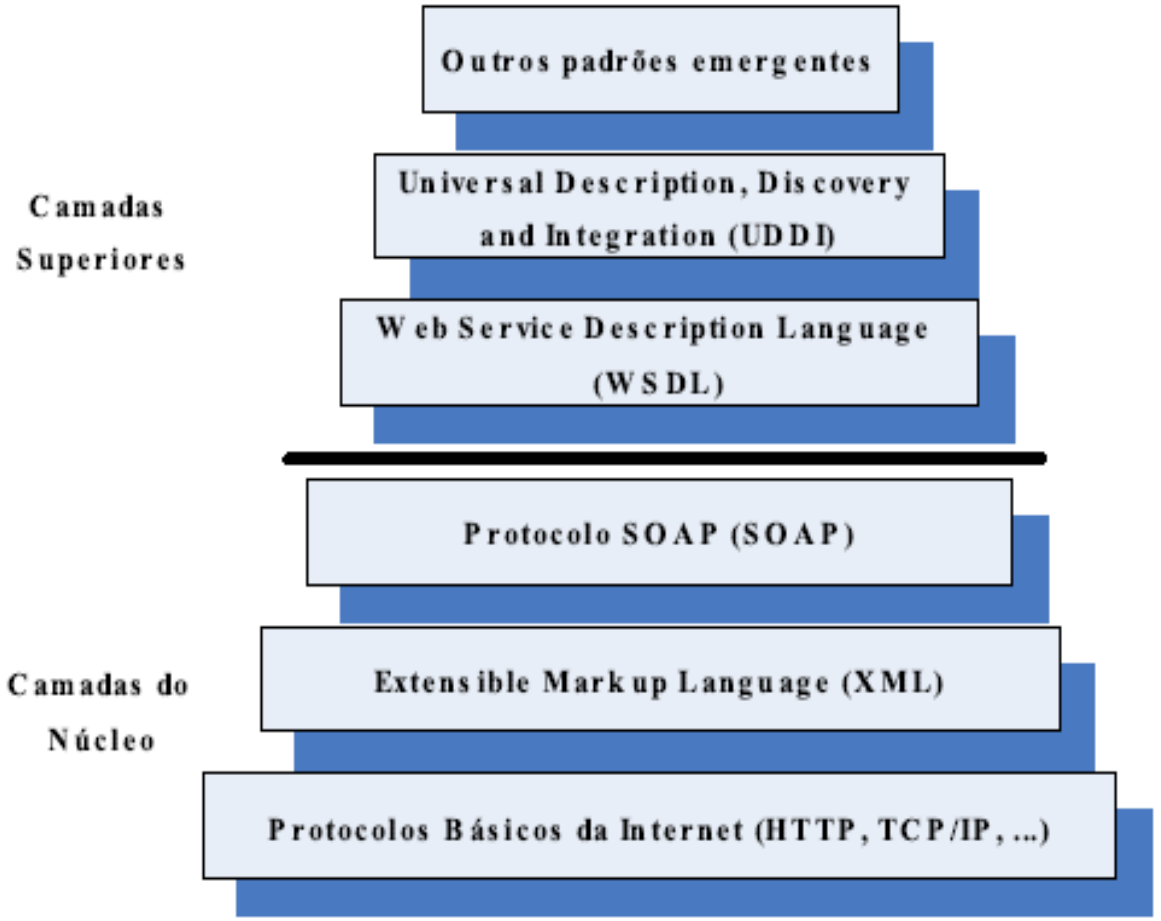
- Web Services são aplicações fracamente acopladas que interagem dinamicamente através de redes TCP/IP (Internet e/ou Intranets), através da publicação, localização e invocação destes pela Web.



Principais Tecnologias Empregadas por Web Services

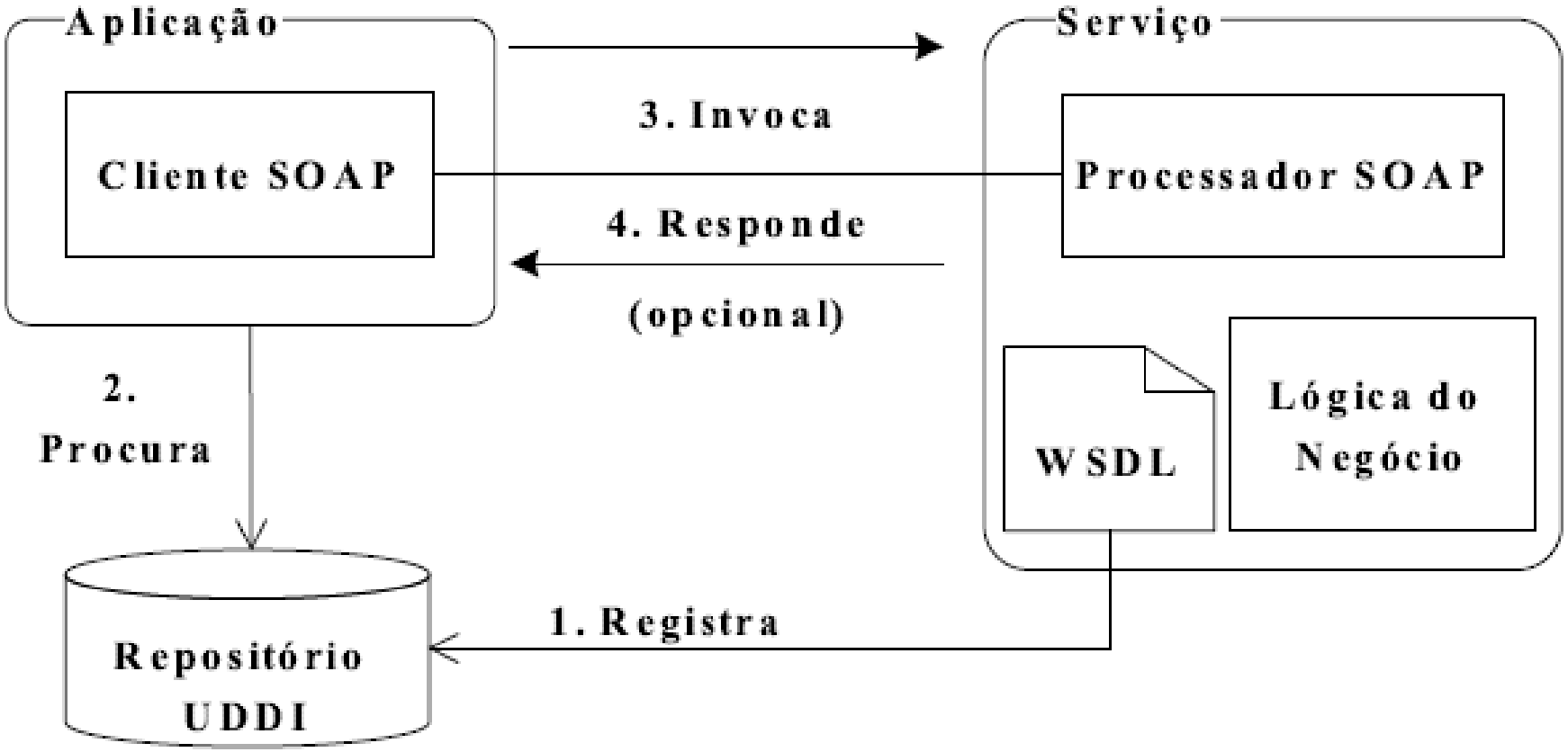
- O núcleo de um Web Service é composto de:
  - um protocolo de Internet (**HTTP**, geralmente), através do qual são enviadas mensagens **XML** envelopadas, isto é, devidamente encapsuladas pelo protocolo **SOAP**.
  - Como camadas superiores (auxiliares, com o intuito de agregar funcionalidades aos Web Services) destacamos a linguagem de descrição dos serviços providos pelo web service - **WSDL** e o repositório no qual podem ser publicados e localizados todos os web services - **UDDI**.

Pilha de Protocolos



- Descoberta
  - UDDI
- Descrição de Serviços
  - WSDL
- Mensagens
  - SOAP
- Transporte
  - HTTP (POST), SMTP, TCP,...

Arquitetura/Protocolos resumido



Interação entre os consumidores, provedores e brokers (intermediadores) de serviços.

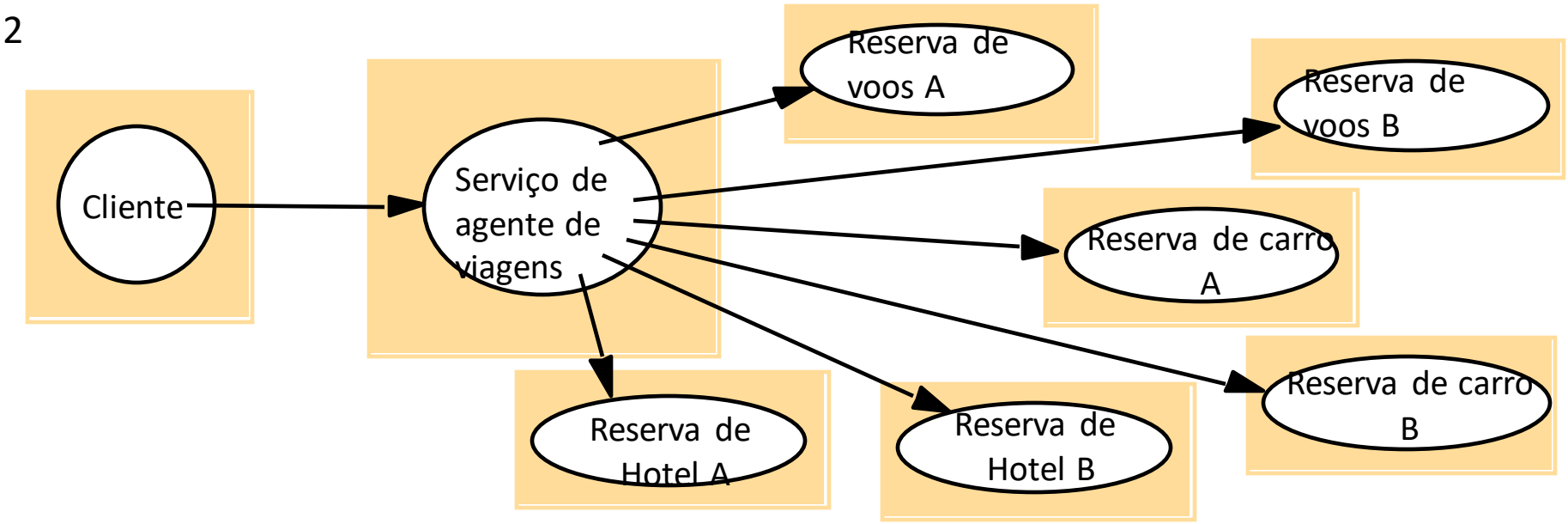
1. Um provedor de serviço registra a descrição de seus serviços através de um arquivo WSDL em um repositório UDDI
2. após, quando um consumidor em potencial (usuário e/ou web service) que está procurando por um serviço com as características descritas no WSDL, acima mencionado, as encontra, é realizado o envio deste arquivo para este cliente.
3. Deste momento em diante, consumidor e provedor são capazes de interagir através de requisições (3) e
4. respostas (4).

Combinação de serviços web

- O fornecimento de uma interface de serviço permite que suas operações sejam combinadas com as de outros serviços para fornecer uma nova funcionalidade.
  - Exemplo: um serviço de agente de viagens:
    - As pessoas utilizam seus navegadores para fazer as reservas individualmente x
    - Uma única interface que que poderia usar vários serviços para fornecer uma combinação de serviços.

O serviço de agente de viagens

Figure 19.2



Ver:

- http://aws.amazon.com/
- http://code.google.com/apis/maps/documentation/elevation/
- http://developer.yahoo.com/search/
- http://developer.ebay.com/

Padrões de comunicação

- Comunicação *síncrona* x assíncrona
  - Requisição-resposta
- Comunicação baseada em eventos
  - Por exemplo, um serviço de diretórios

Outras características

- Nenhuma modelo de programação em particular;
- Representação de mensagens via XML;
- Referências de serviços via URL;
- Ativação de serviços
  - Continuamente x sob demanda
- Transparência
  - *Middleware*
  - *Proxies (stubs)*

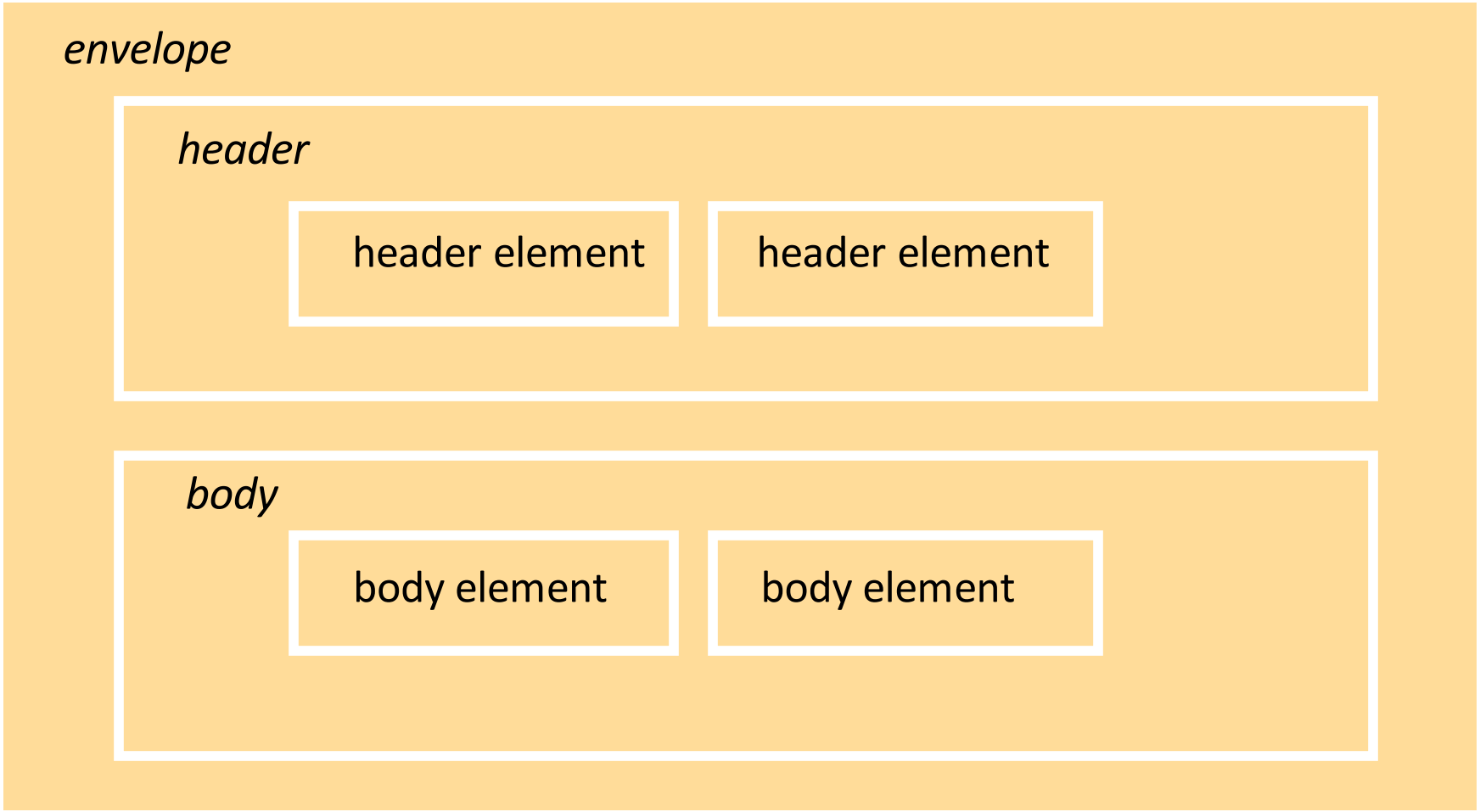
Simple Object Access Protocol (SOAP)

- O Especificação do SOAP declara:
  - Como usar a XML para representar o conteúdo das mensagens individuais
  - Regras de como os destinatários devem processar o XML
  - Como HTTP, SMTP, TCP e UDP são usados para *transportar* mensagens SOAP
  - Como usar o método POST para mensagens de requisição/reposta
  - APIs implementadas em muitas linguagens, incluindo Java, Java Script, Perl, Python, .NET, C, C++, C#, Visual Basic e etc.

Simple Object Access Protocol

- Envelope: encapsula uma mensagem SOAP
  - Cabeçalho(opcional)
    - Estabelece o contexto do serviço
    - Log ou auditoria de das operações
    - Um intermediário pode interpretar e atuar sobre as informações
  - Corpo
    - Encapsula um documento XML sobre um serviço web em particular
    - Os remetentes usam as descrições de serviço para gerar o corpo e para garantir que ele possua o conteúdo correto. Os destinatários as utilizam para analisar e validar o conteúdo.

Mensagem SOAP em um envelope

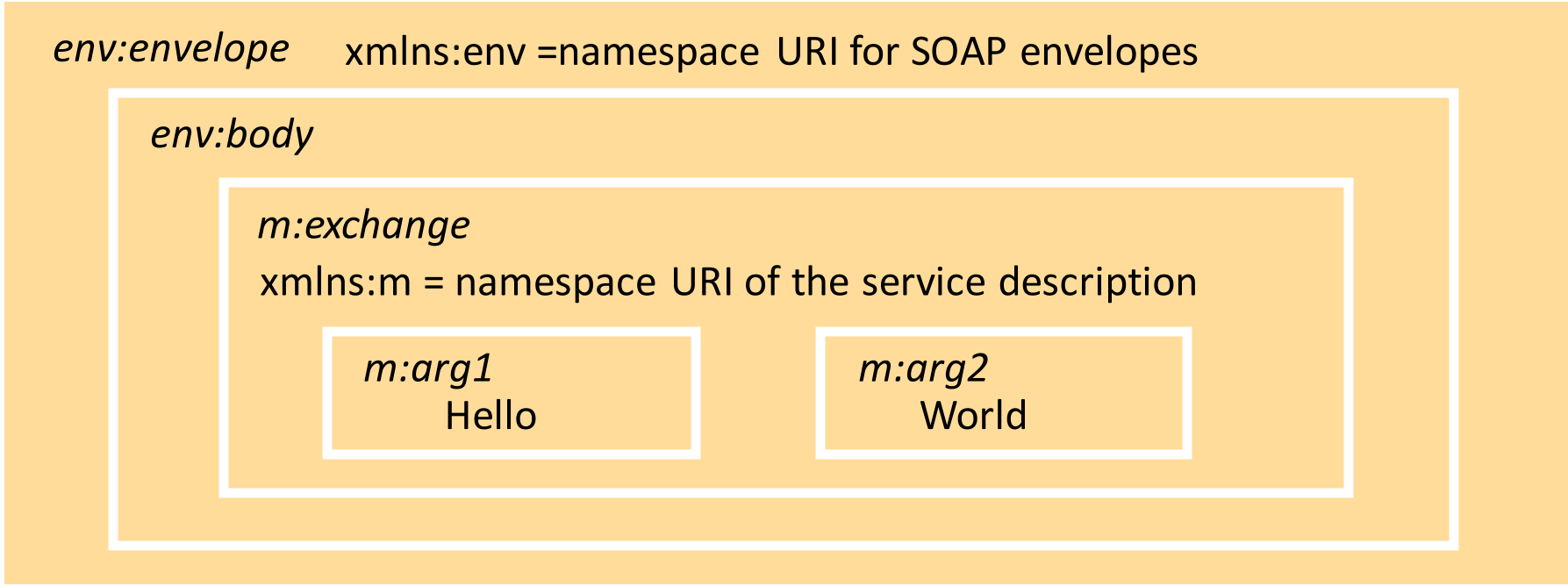


Esqueleto da mensagem SOAP

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

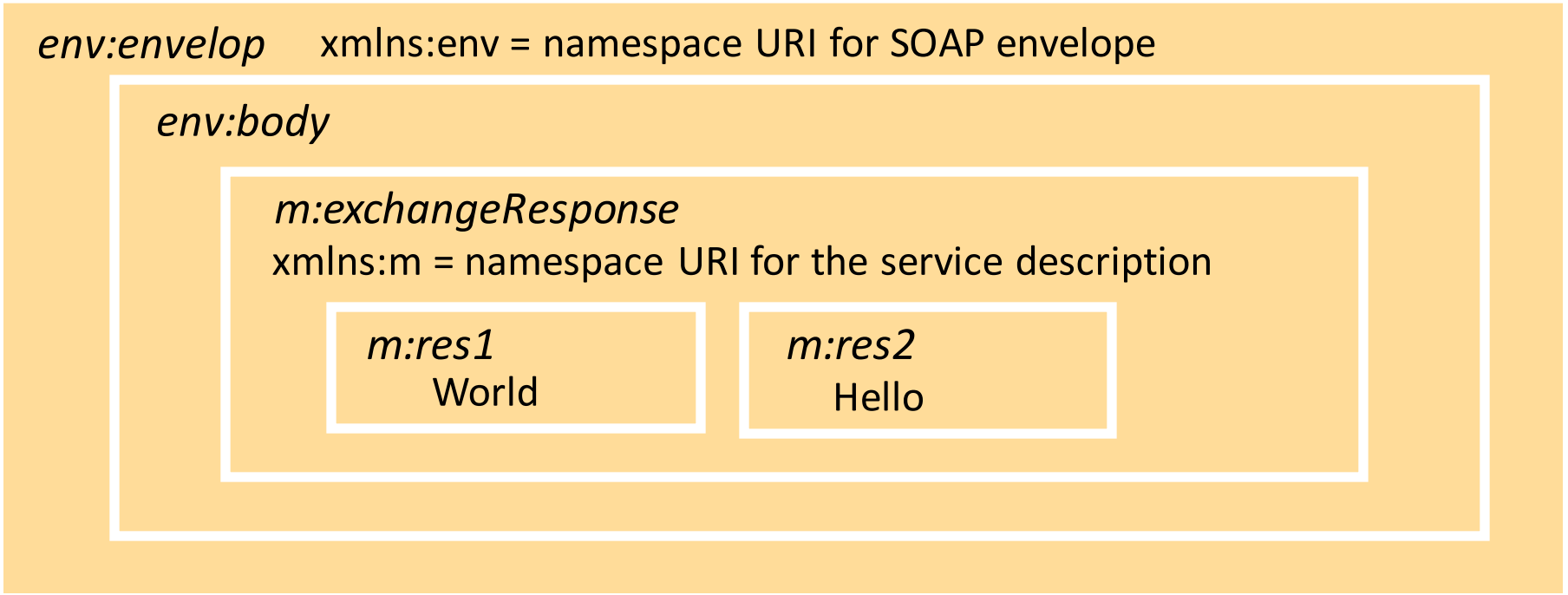
Exemplo de uma requisição simples sem cabeçalhos

Figure 19.4



Obs.: Nesta figura e na próxima, cada elemento XML é representado por uma caixa sombreada com seu nome em itálico, seguido por quaisquer atributes e seu conteúdo

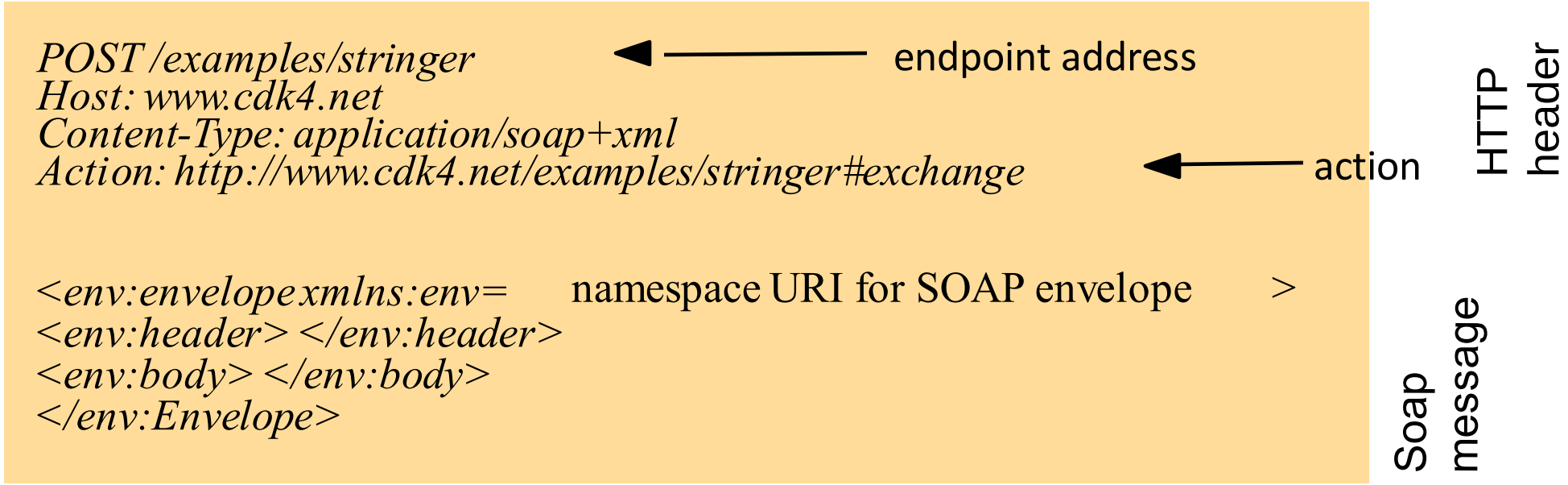
Exemplo de resposta correspondente à requisição da Figura 19.4



Transporte de mensagens SOPA

- Os cabeçalhos HTTP especificam o endereço do ponto final e ação a ser executada;
- O parâmetro *Action* se destina a otimizar o envio, revelando o nome da operação;
  - Permite que o módulo correto seja escolhido sem inspecionar a mensagem SOAP. Caso, o Contente-Type: application/soap+xml
- O corpo HTTP transporta a mensagem SOAP.

Figure 19.6 Use of HTTP POST Request in SOAP client-server communication



Objetos Distribuídos vs Web Services

- Uma URL não pode ser comparada a uma referencia remota, pois ela aponta para um único objeto.
- Serviços WEB não criam instâncias de objetos.
- JAX-RPC:
  - desenvolvedores sabem que não estão usando invocação remota transparente de Java para-Java, mas sim usando modelo de serviços web, no qual objetos remotos não podem ser instanciados.
- Referencias de objetos remotos não podem ser passadas como argumentos nem retornadas como resultado.
- Não suportam serventes
  - As implementações de interfaces não devem ter construtores nem métodos principais.

Interface de serviço web java *ShapeList*

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

SOAP com JAVA

- JAX-RPC
  - Oculta os detalhes do SOAP para programadores, clientes e serviços
  - Mapeia alguns dos tipos da linguagem JAVA para definições em XML (Mensagens SOAP e WSDL)
    - *Integer, String, Date, Calendar, URI, Vector, Arrays,* dentre outros.
  - Instâncias de classes podem ser passadas como argumento ou resultado desde que:
    - Seus atributos sejam dos tipos suportados;
    - Não implementem *Remote;*
    - Tenham um construtor público padrão.

SOAP com JAVA

- JAX-RPC
  - Interface de Serviço
    - Deve estender *Remote.*
    - Não deve ter declarações constantes, como *public final static.*
    - Métodos devem lançar *RemoteException* ou uma subclasse.
    - Parâmetros e tipos de retorno dos métodos devem ser dos tipos suportados pelo JAX-RPC

Figure 19.8 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) {}
}
```

Programa Servidor

- Não há método ***main***(), nem **constrututor**.
  - Serviço web é um **objeto único** que oferece um conjunto de procedimentos
- A interface e sua implementação são compiladas normalmente (***javac***).
- ***wscompile*** e ***wsdeploy*** são usados para gerar a classe **esqueleto** e a descrição do serviço (em **WSDL**), usando informações relativas ao URL do serviço, seu nome, e sua descrição em XML.
- O nome do serviço (*MyShapeListService*) é usado para gerar o nome da classe do programa cliente que irá acessá-lo (*MyShapeListService_impl*)

Programa Servidor (2)

- Contêiner de *servlet*
  - A implementação do serviço é executa como um *servlet* dentro de contêiner, cuja função é carregar, inicializar e executar *servlets.*
  - Inclui despachante e esqueleto
    - Quando chega a requisição, o despachante seleciona o esqueleto relacionado, o qual **a transforma** em código Java e passa para o método apropriado do *servlet.*
    - O resultado é retornado ao esqueleto, que o transforma novamente em mensagem SOAP
  - Serviço é acessado via URL

Programa Cliente

- Acessa o serviço através de *proxies* ou *stubs* (estáticos ou dinâmicos) ou interface de invocação dinâmica.
  - *Proxies* estáticos são gerados pelo *wscompile* a partir da descrição do serviço
    - Nome da classe proxy é gerada pela concatenação do nome do serviço com _Impl
  - *Proxies* dinâmicos são gerados em tempo de execução

Proxy estático

Figure 19.9 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();
                proxy._setProperty
                    (javax.xml.rpc.Stub.ENDPOINT_ADDRESS_PROPERTY, args[0]);
                ShapeList aShapeList = (ShapeList)proxy;
                GraphicalObject g = aShapeList.getAllState(0);
            } catch (Exception ex) { ex.printStackTrace(); }
        }
        private static Stub createProxy() {
                return
                    (Stub) (new MyShapeListService_Impl().getShapeListPort());
    }
}
```

Implementação SOAP Java (1)

- *Semelhante* à arquitetura RMI com exceção do *módulo de referência remota*
- Módulo de Comunicação
  - Módulos HTTP no cliente e no servidor. O módulo do servidor seleciona o despachante de acordo com o URL dado no cabeçalho de ação HTTP.
- *Proxy* ou *Stub* Cliente
  - Conhece a URL do serviço e empacota nome do método e seus argumentos, além de uma referencia para o esquema XML do serviço em um envelope SOAP

Implementação SOAP Java (2)

- Despachante e Esqueleto
  - Localizados no contêiner de *servlet*
  - Despachante extrai o nome da operação do cabeçalho de ação (requisição HTTP) e invoca o método correspondente no esqueleto apropriado, passando o envelope SOAP
  - Esqueleto analisa o envelope, extrai os argumentos, chama o método correspondente e monta um envelope de resposta SOAP com o resultado

Implementação SOAP Java (3)

- Erros, falhas e correção
  - Relatados pelo Módulo HTTP, pelo despachante e pelo esqueleto e pelo serviço
  - Serviço
    - Valor de retorno ou parâmetros de falha especificados na declaração do serviço
  - Esqueleto
    - Verifica se há uma requisição no envelope e se o XML está bem formado
    - Verifica se a operação existe e se os argumentos foram passados corretamente
    - Verificações semelhantes são feitas pelo proxy no lado cliente

WSDL - *Web Service Description Language*

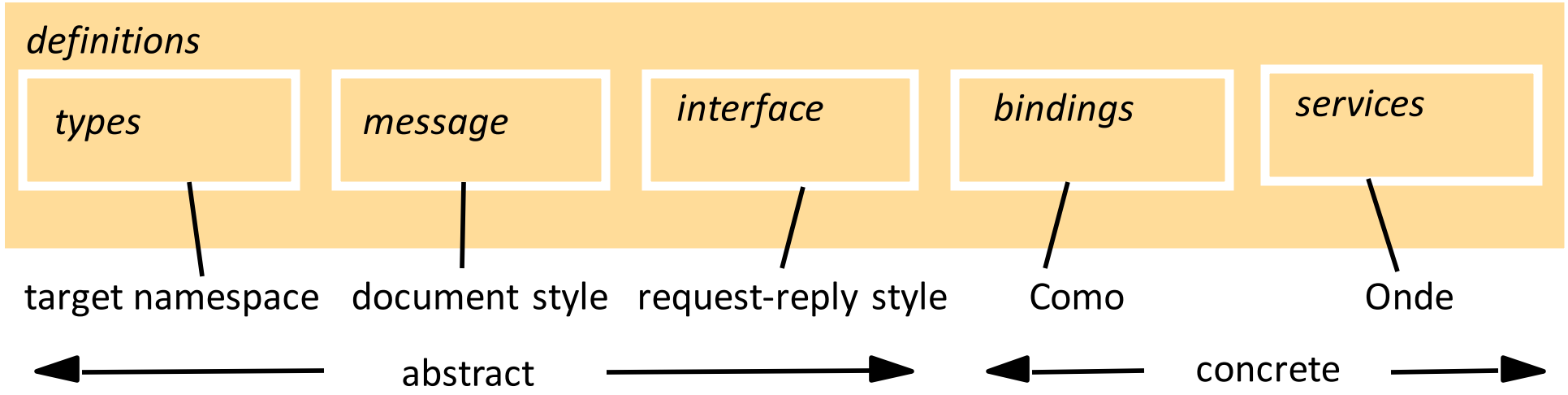
- WSDL é uma linguagem utilizada para descrever *Web Services* definida pelo W3C.
- A descrição em WSDL de um Web Service fornece a especificação da interface deste Web Service, ou seja, o que cada serviço faz e como invocá-los.
- É baseado em XML (como o SOAP)

WSDL - *Web Service Description Language*

- Outro aspecto importante nos documentos WSDL é a possibilidade de geração de código (*stubs*, *skeletons* e parte do código do cliente e/ou servidor) a partir de uma definição WSDL.
- Normalmente um arquivo WSDL é gerado automaticamente pela ferramenta de desenvolvimento de software utilizada na criação do Web Service.
  - Ex: WSDL4J

Os principais elementos em uma descrição WSDL

- A WSDL separa a parte abstrata de uma descrição de serviço da parte concreta, como se vê na figura abaixo.



Os principais elementos em uma descrição WSDL

- A parte abstrata inclui um conjunto de definições dos tipos usados pelo serviço, i.e., os tipo dos valores trocados nas mensagens;
- A parte concreta especifica como e onde o serviço pode ser contatado;
- A modularidade é inerente de uma definição WSDL;
  - As definições podem ser referenciadas a partir de vários documentos WSDL diferentes;

WSDL - *Web Service Description Language*

- Types
  - Definem os tipos de dados enviados/recebidos
- Message
  - Define os tipos de mensagens enviadas/recebidas
- Interface
  - Define de forma abstrata o conjunto de operações fornecido pelo serviço
- Binding
  - Como o serviço é acessado – Protocolos
  - Formatos de mensagens, representação externa, transporte
- Service
  - Onde acessar o serviço (URL)

Mensagens e Operações

- Definem como se dará as requisições e as respostas, i.e., como as mensagens serão trocadas;
- Mensagens de requisição e resposta WSDL para a operação *newShape*

```
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

Figure 19.11

Interface

- O conjunto de operações pertencentes a um serviço web é agrupado em um elemento da XML chamado *interface,* às vezes também chamado de *portType*.
- As opções disponíveis são:
  - In-Out, In-Only, Robust In-Only,
  - Out-In, Out-Only e Robust Out-Only
- Herança: toda interface WSDL pode estender uma ou mais outras interfaces WSDL. Assim, uma interface suporta as operações de todas as interfaces que ela estende.

Padrões de troca de mensagem para operações

Name	*Messages sent by*			
	Client	*Server*	*Delivery*	*Fault message*
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

Figure 19.12

Interface

- Exemplo, operação *newShape*:

Figure 19.13 WSDL operation newShape

operation name = "newShape "
pattern = In-Out

input message = "tns:ShapeList_newShape "

output message = "tns:ShapeList_newShapeResponse"

tns – target namespace xsd – XML schema definitions

Os nomes *operation*, *pattern, input,e output* são definidos no esquema XML da WSDL

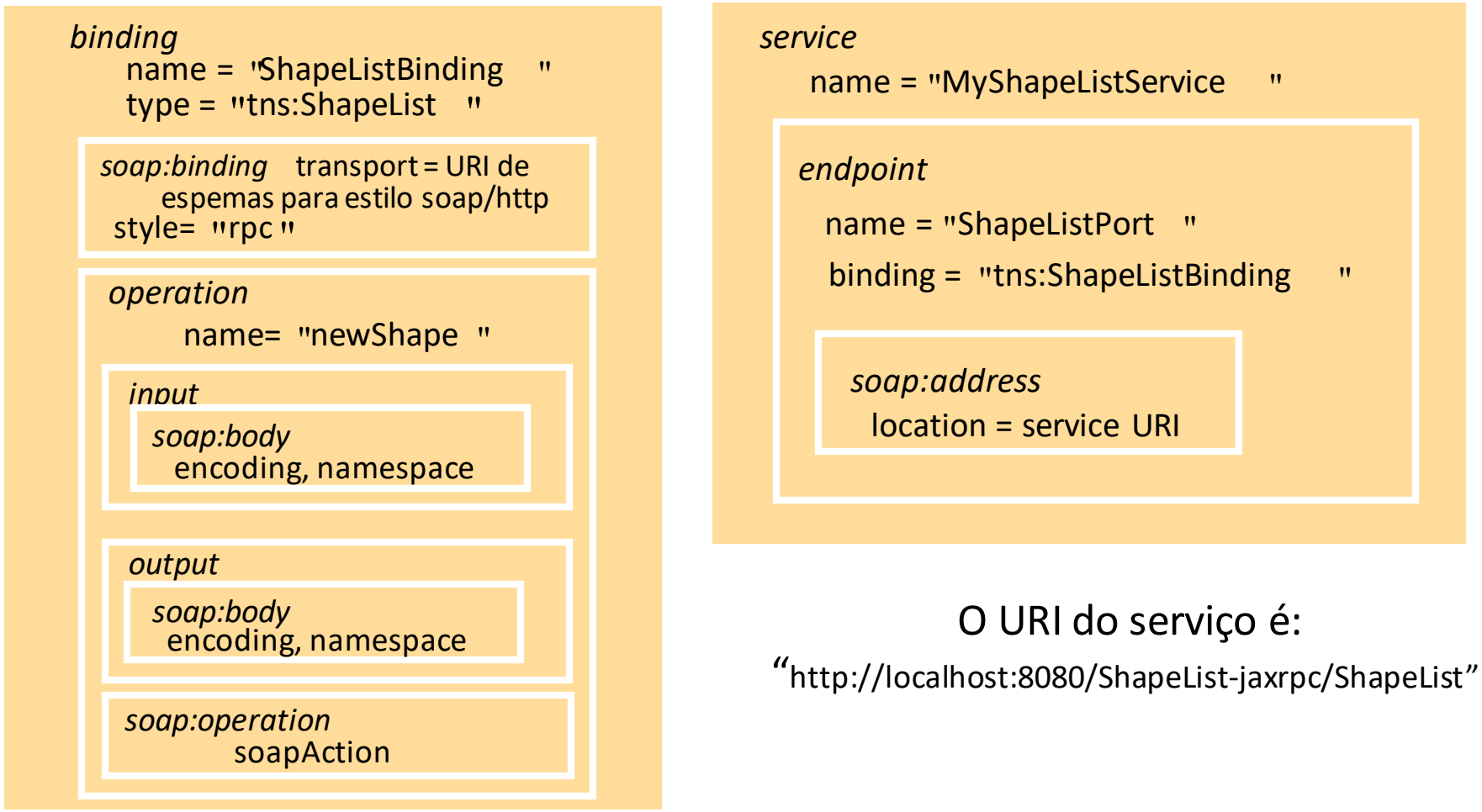
Parte concreta (Vínculo)

- A parte restante (concreta) de um documento WSDL consiste em *binding* (a escolha de protocolos) e *service* (a escolha do endereço do ponto final ou do servidor).
  - Ambas estão relacionadas.
- Vínculo: formato das mensagens e representação externa de dados, por exemplo: SOAP, HTTP e MIME.
  - Padrão de troca de mensagem: *rpc* (req-res) ou *document* (padrão);
  - Esquema XML do formato da mensagem: SOAP;
  - Esquema XML da representação externa de dados: SOAP.

Parte concreta (Serviço)

- Cada elemento *service* especifica o nome do serviço e pontos finais.
- Um vínculo SOAP, utiliza um elemento *soap:address* para especificar o URI da localização do serviço.
- Documentação: pode ser inserida no elemento *documentation* na maioria dos pontos de um WSDL.

Figure 19.14 SOAP binding and service definitions



O URI do serviço é:

"http://localhost:8080/ShapeList-jaxrpc/ShapeList"

UDDI - Universal Description, Discovery, and Integration

- O UDDI consiste em um serviço de nomeação e localização de *web services* estruturado na forma de repositórios.
- Após um *web service* ser modelado/desenvolvido é feita a publicação deste num repositório UDDI.
- A partir desse momento as informações necessárias para a localização e a utilização do serviço disponibilizado por este *web service* se tornam acessíveis para os clientes na forma de um arquivo WSDL.
- WSDL pode ser buscado pelo nome (como lista telefônica) ou por atributos (como páginas amarelas)

UDDI - Universal Description, Discovery, and Integration

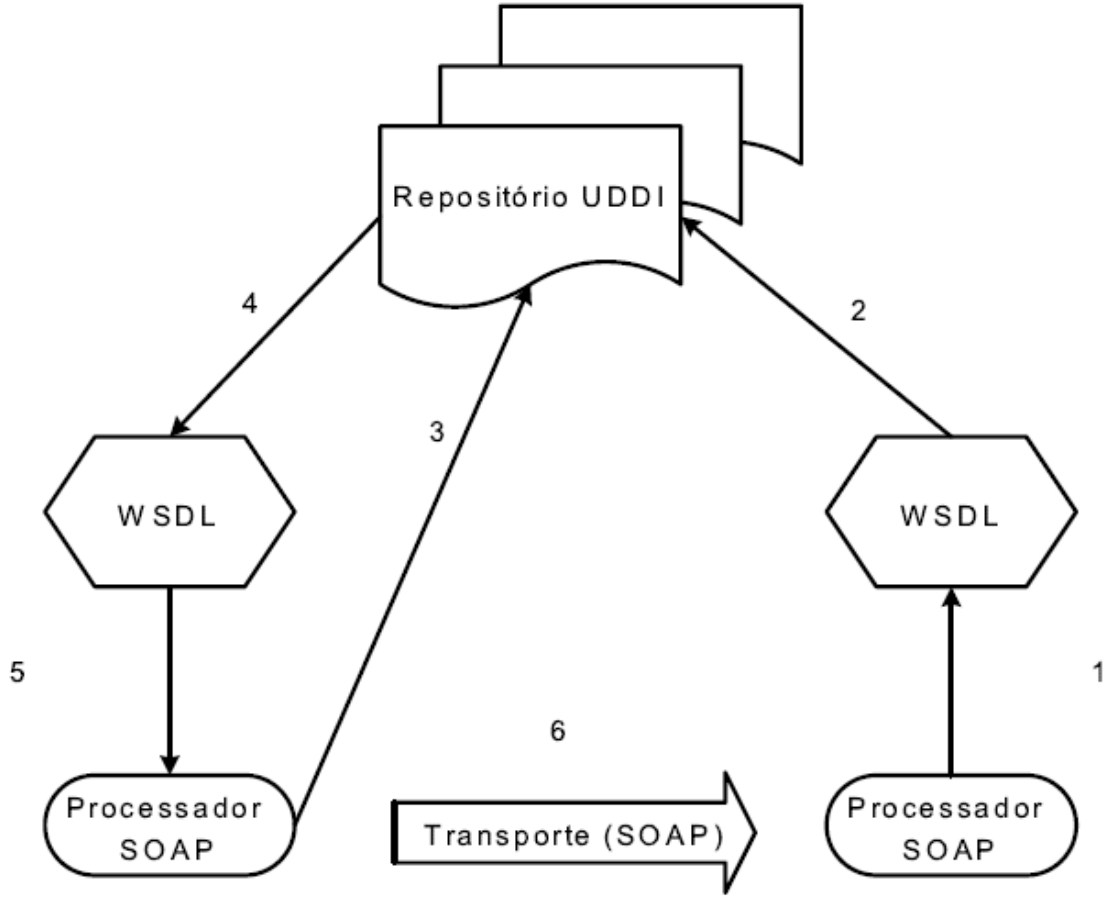


Figure 19.15 The main UDDI data structures