

Introdução à Dados Semiestruturados e XML

Vanessa Braganholo
{vanessa@ic.uff.br}

Roteiro da Aula

- ▶ Dados Semiestruturados
- ▶ O que é XML
- ▶ XML x HTML
- ▶ Terminologia XML



Dados estruturados ou não...

- ▶ **Dados estruturados**

- ▶ Estrutura é conhecida a priori

Ex.: Dados de um SGBD relacional têm um esquema relacional associado

- ▶ **Dados não estruturados**

- ▶ Não há nenhuma estrutura prévia

Ex.: imagem, video, áudio, etc.



Dados Semiestruturados

- ▶ Dados irregulares
- ▶ Dados incompletos
- ▶ Não necessariamente está de acordo com um esquema



Dados Irregulares

- ▶ Livros podem ser descritos por uma estrutura de partes e capítulos ou podem ser descritos somente por capítulos
- ▶ A descrição de uma disciplina pode variar em termos de atributos de um departamento para outro
 - ▶ faltam atributos ou possuem atributos a mais



Dados Incompletos

- ▶ Nem todo endereço tem caixa postal
- ▶ Nem todo livro tem apêndice ou prefácio



Esquema Opcional

- ▶ Sua estrutura não é previamente conhecida, pode não existir à parte
- ▶ São auto-descritivos, i.e., embute a própria estrutura



Auto-descrição

- ▶ pares atributo-valor

{name: “John Smith”, tel: 3456, age: 32}

- ▶ valor de atributo pode também conter estrutura

{name: {first: “John”, last: “Smith”}, tel: 3456, age: 32}

- ▶ rótulos de atributo não necessariamente únicos

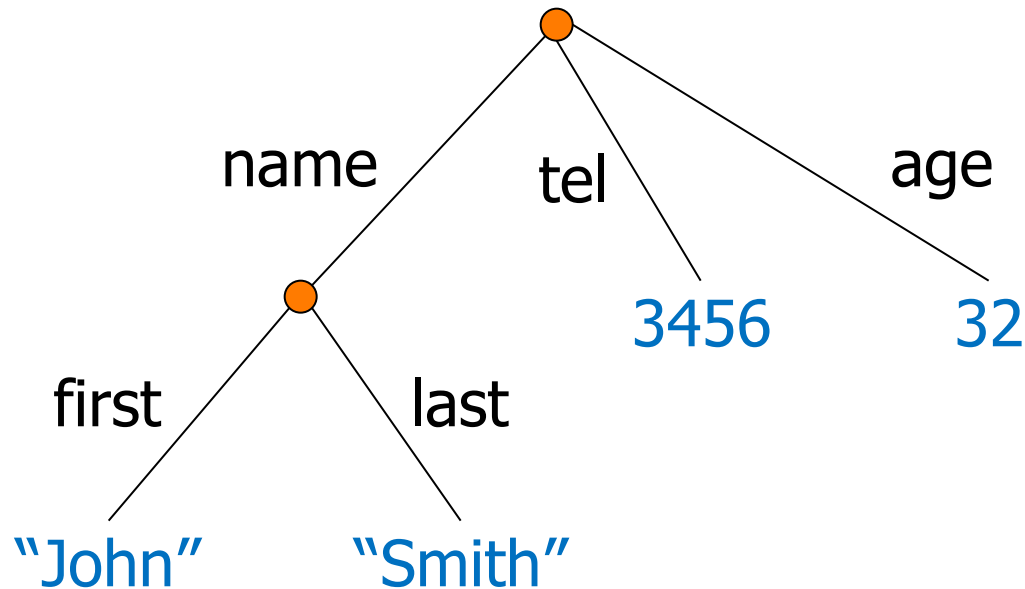
{name: “John Smith”, tel: 3456, tel: 7891}



Representação gráfica

- ▶ nós representam objetos conectados por arestas que os descrevem

Ex.: {name: {first: "John", last: "Smith"}, tel: 3456, age: 32}



Situações típicas

- ▶ Quando os dados não podem ser restritos a um esquema
 - ▶ Difícil definir uma estrutura... Ex: contratos
- ▶ Quando não há compromisso com o conteúdo
 - ▶ Pode-se ter muitos dados faltando... Ex. Leis
- ▶ Quando as fontes de dados são heterogêneas e é preciso integrar dados...
 - ▶ Descrições equivalentes mas distintas...



Exemplos

▶ Arquivos BibTeX

- ▶ Têm estrutura, mas ela não é regular
 - ▶ Alguns atributos não aparecem, apesar de obrigatórios

```
@article{Gettys90,  
  author = {Jim Gettys and Phil Karlton and Scott McGregor},  
  title = {The {X} Window System, Version 11},  
  journal = {Software Practice and Experience},  
  volume = {20},  
  number = {S2},  
  year = {1990},  
  postscript = "papers/gettys90.ps.gz",  
  abstract = {A technical overview of the X11 functionality}  
}
```

Arquivos GenBank

```
LOCUS      SCU49845      5028 bp      DNA      PLN      21-JUN-1999
DEFINITION Saccharomyces cerevisiae TCP1-beta gene, partial cds, and Axl2p
            (AXL2) and Rev7p (REV7) genes, complete cds.
ACCESSION  U49845
VERSION    U49845.1  GI:1293613
KEYWORDS   .
SOURCE     Saccharomyces cerevisiae (baker's yeast)
  ORGANISM Saccharomyces cerevisiae
            Eukaryota; Fungi; Ascomycota; Saccharomycotina; Saccharomycetes;
            Saccharomycetales; Saccharomycetaceae; Saccharomyces.
REFERENCE  1  (bases 1 to 5028)
  AUTHORS  Torpey,L.E., Gibbs,P.E., Nelson,J. and Lawrence,C.W.
  TITLE    Cloning and sequence of REV7, a gene whose function is required for
            DNA damage-induced mutagenesis in Saccharomyces cerevisiae
  JOURNAL  Yeast 10 (11), 1503-1509 (1994)
  MEDLINE  95176709
  PUBMED   7871890
FEATURES   Location/Qualifiers
            CDS           <1..206
                       /codon_start=3
                       /product="TCP1-beta"
                       /protein_id="AAA98665.1"
                       /db_xref="GI:1293614"
                       /translation="SSIYNGISTSGLDLNNGTIADMRQLGIVESYKLRVSSASEA
            AEVLLRVDNIIIRARPRTANRQHM"
            gene          687..3158
                       /gene="AXL2"
```

Exemplos

- ▶ Guia de restaurantes (Palo Alto Weekly newspaper)
 - ▶ Cada restaurante apresenta uma estrutura diferente

Guide

Restaurant

Name "Blues on the Bay"

Category "Vegeterian"

Entree

Name "Black bean soup"

Price "10.00"

Entree

Name "Asparagus Timbale"

Price "22.50"

Location

Street "1890 Wharf Ave"

City "San Francisco"

Restaurant

Name "McDonald's"

Category "Fast Food"

Price "cheap"

Nearby "Blues on the Bay"

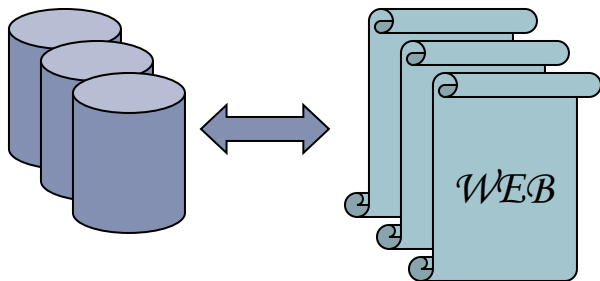
Arquivo JSON

```
{ "Aluno" : [  
  { "nome": "João", "nota": [ 8, 9, 7 ] },  
  { "nome": "Maria", "nota": [ 8, 10, 7 ] },  
  { "nome": "Pedro", "nota": [ 10, 10, 9 ] }  
]  
}
```

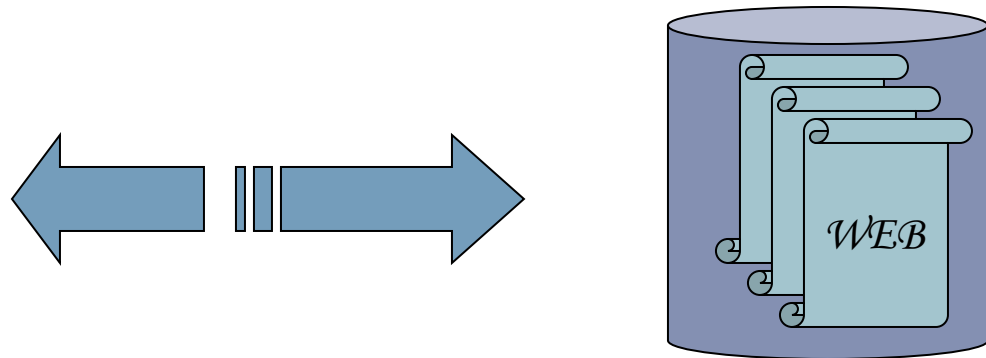
Web: grande fonte de dados semiestruturados

- ▶ **Páginas web contém informação valiosa**
 - ▶ Documentos de conteúdo importante
 - ▶ Dados armazenados em BD's disponibilizados na web

Antes, a web era vista como uma forma de disponibilizar informação e/ou sistemas.

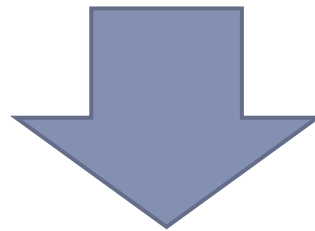


Hoje, a Web é vista como um grande banco de dados.



Descrever os dados da Web

- ▶ Tratar dados semiestruturados
- ▶ Separar o conteúdo:
 - ▶ Independência de armazenamento
 - ▶ Permite a visualização de dados provenientes de fontes heterogêneas
 - ▶ Independência de apresentação
 - ▶ Permite que as aplicações apresentem/tratem os dados como lhes é conveniente



XML

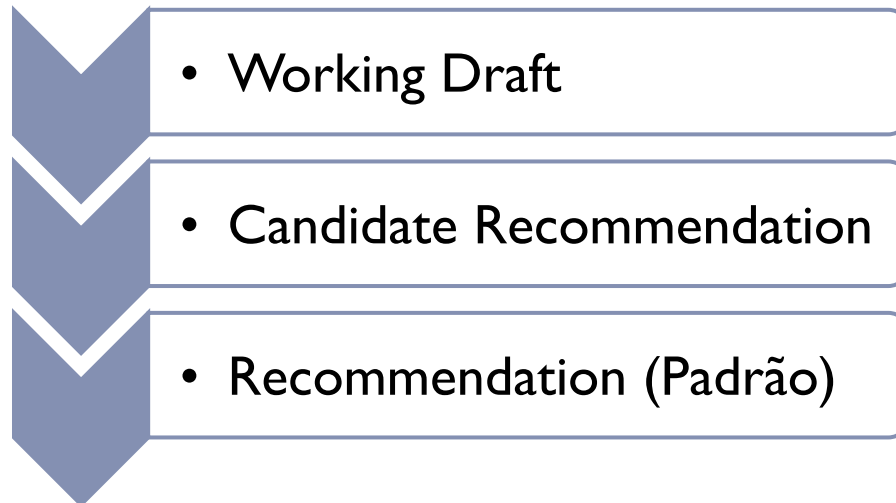
O que é XML?

- ▶ XML = eXtensible Markup Language
- ▶ Padrão para marcação de dados na Web, com foco na descrição do conteúdo



W3C – www.w3.org

- ▶ Órgão responsável pela padronização de iniciativas ligadas à Web
 - ▶ Ex.: HTML, XML e iniciativas relacionadas, entre outros
- ▶ Especificações dessas iniciativas são classificadas de acordo com seu nível de “maturidade”



HTML x XML

- ▶ HTML – descreve o **formato** do documento
 - ▶ HTML tem um conjunto fixo de tags e não descreve conteúdo
- ▶ XML – descreve o **conteúdo** do documento
 - ▶ Usuário define suas próprias tags para criar uma estrutura
 - ▶ Um documento XML não tem nenhuma instrução para apresentação



Histórico: XML

- ▶ 1993: primeiros trabalhos sobre adaptação das técnicas SGML à Web (Sperberg).
- ▶ ‘HTML to the Max: A Manifesto for Adding SGML Intelligence to the World Wide Web’

- ▶ Junho 1996: criação de um grupo de trabalho no W3C



Histórico

▶ 1996

- ▶ 80 peritos em SGML uniram forças ao W3C (World Wide Web Consortium)
- ▶ Objetivo: definir uma linguagem de marcação com o poder da SGML, porém fácil de ser implementada
- ▶ influência do LOREL

▶ 10 fevereiro 1998

- ▶ publicação da recomendação para versão 1.0 da linguagem



SGML - Características

- ▶ “Standard Generalized Markup Language”
- ▶ Uma linguagem de marcação abrangente mas complexa
- ▶ Desenvolvida por Charles F. Goldfarb
- ▶ Adequada para aplicações envolvendo documentos grandes e complexos
- ▶ Tornou-se um padrão ISO (ISO 8879) na década de 80



SGML e XML

- ▶ SGML - norma ISO 8879:1986
- ▶ SGML vinha sendo utilizada na indústria no suporte à técnicas de documentação
- ▶ Muito complexa para utilização de «público em geral»
- ▶ XML usa **10%** de SGML para representar de forma eficaz **90%** dos documentos



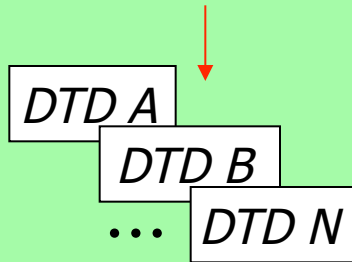
Linguagens de marcação

SGML – linguagem de marcação com regras para definição de classes de documentos



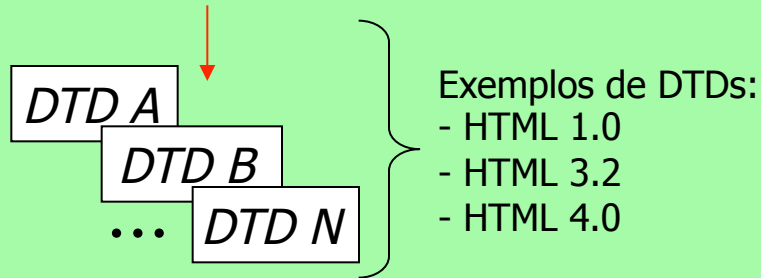
Linguagens de marcação

*SGML – linguagem de marcação com regras para definição de **classes** de documentos*



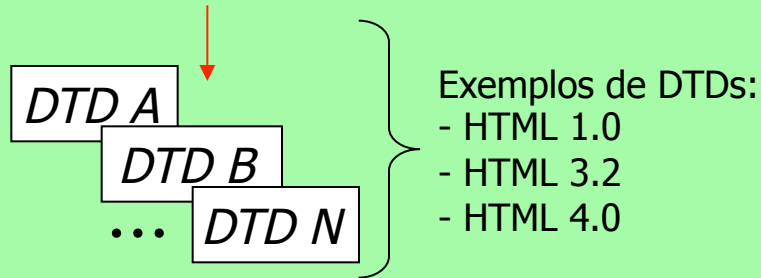
Linguagens de marcação

*SGML – linguagem de marcação com regras para definição de **classes** de documentos*



Linguagens de marcação

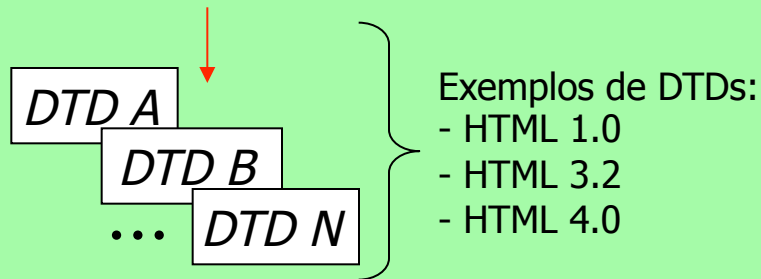
*SGML – linguagem de marcação com regras para definição de **classes** de documentos*



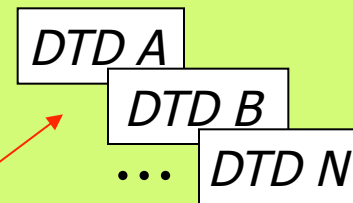
XML - subconjunto da SGML – linguagem de marcação com regras para definição de classes de documentos

Linguagens de marcação

*SGML – linguagem de marcação com regras para definição de **classes** de documentos*



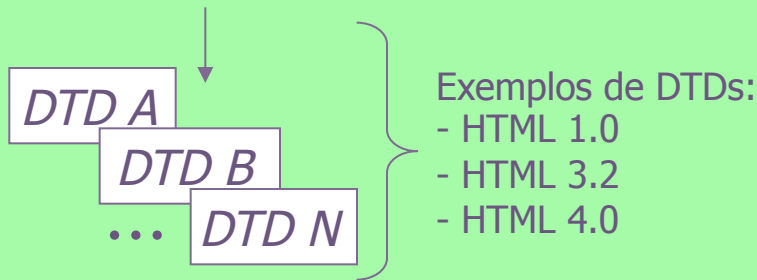
Exemplos de DTDs:
- XHTML 1.0
- DocBook



*XML - subconjunto da SGML – linguagem de marcação com regras para definição de **classes** de documentos*

Linguagens de marcação

SGML – linguagem de marcação com regras para definição de classes de documentos



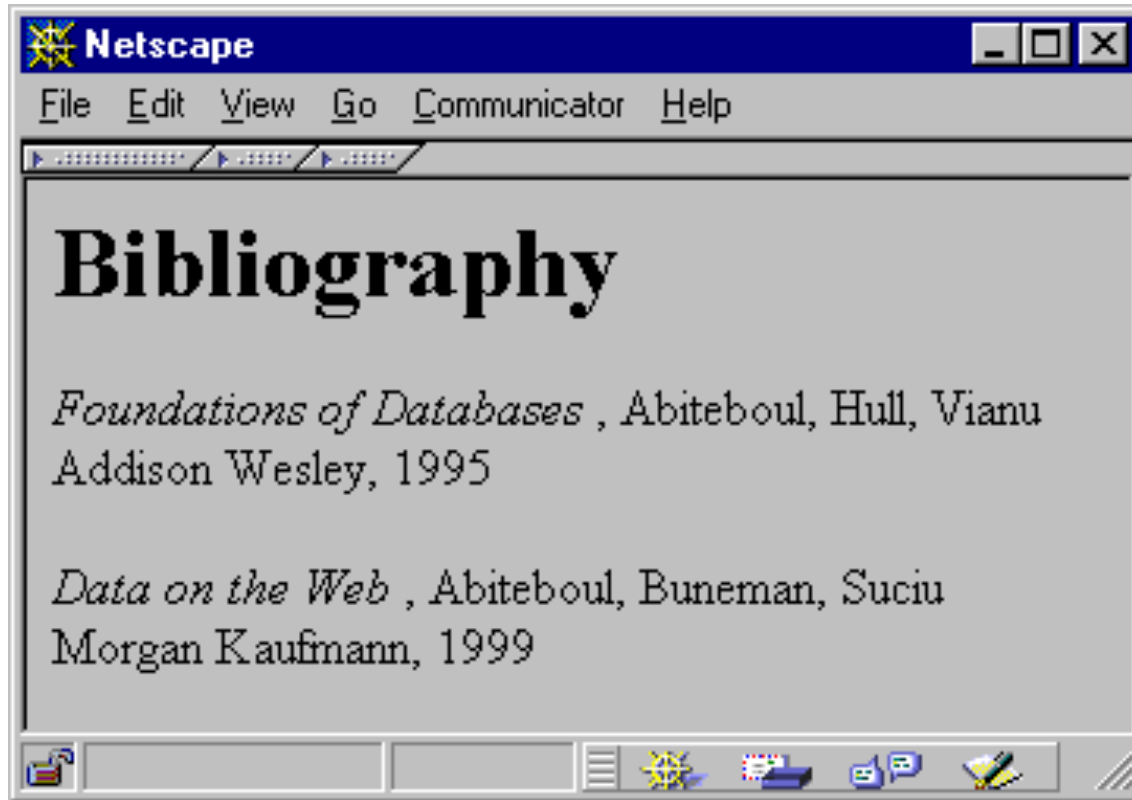
XML é compatível com SGML

Exemplos de DTDs:
- XHTML 1.0
- DocBook



XML - subconjunto da SGML – linguagem de marcação com regras para definição de classes de documentos

De HTML para XML...



HTML descreve a **apresentação!**

Fonte HTML

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
  <HEAD><TITLE>A bibliography on Databases</TITLE>
  <META content="text/html; charset=windows-1252" http-
equiv=Content-Type>
  <META content="MSHTML 5.00.2314.1000" name=GENERATOR>
</HEAD>
<BODY>
  <h1> Bibliography </h1>
  <p> <i> Foundations of Databases </i> Abiteboul, Hull, Vianu <br>
  Addison Wesley, 1995
  <p> <i> Data on the Web </i> Abiteoul, Buneman, Suciu <br>
  Morgan Kaufmann, 1993
</BODY>
</HTML>
```

HTML: Conjunto pré-definido de elementos (tags) para especificação das dimensões de estrutura e apresentação de um documento

Fonte XML

XML: Elementos (tags) definidos pelo usuário da linguagem e servindo para descrever o conteúdo e a estrutura.

```
<bibliography>
  <book>
    <title> Foundations... </title>
    <author> Abiteboul </author>
    <author> Hull </author>
    <author> Vianu </author>
    <publisher> Addison Wesley </
publisher>
    <year> 1995 </year>
  </book>
  ...
</bibliography>
```

XML descreve o conteúdo!!!



Sintaxe XML

XML (<http://www.w3.org/TR/xml/>)

▶ Documento XML

- ▶ Seqüência de elementos que englobam texto ou outros elementos
- ▶ Elementos podem conter atributos

XML

▶ Elemento

- ▶ Delimitado por marcas (*tags*)
- ▶ Possuem uma marca inicial e uma marca final
- ▶ Tudo o que estiver delimitado por essas marcas faz parte do conteúdo do elemento
- ▶ Ex: `<empregado>João</empregado>`

▶ Atributo

- ▶ Podem aparecer dentro da marca **inicial** de um elemento
- ▶ Ex: `<empregado cod="E01">João</empregado>`

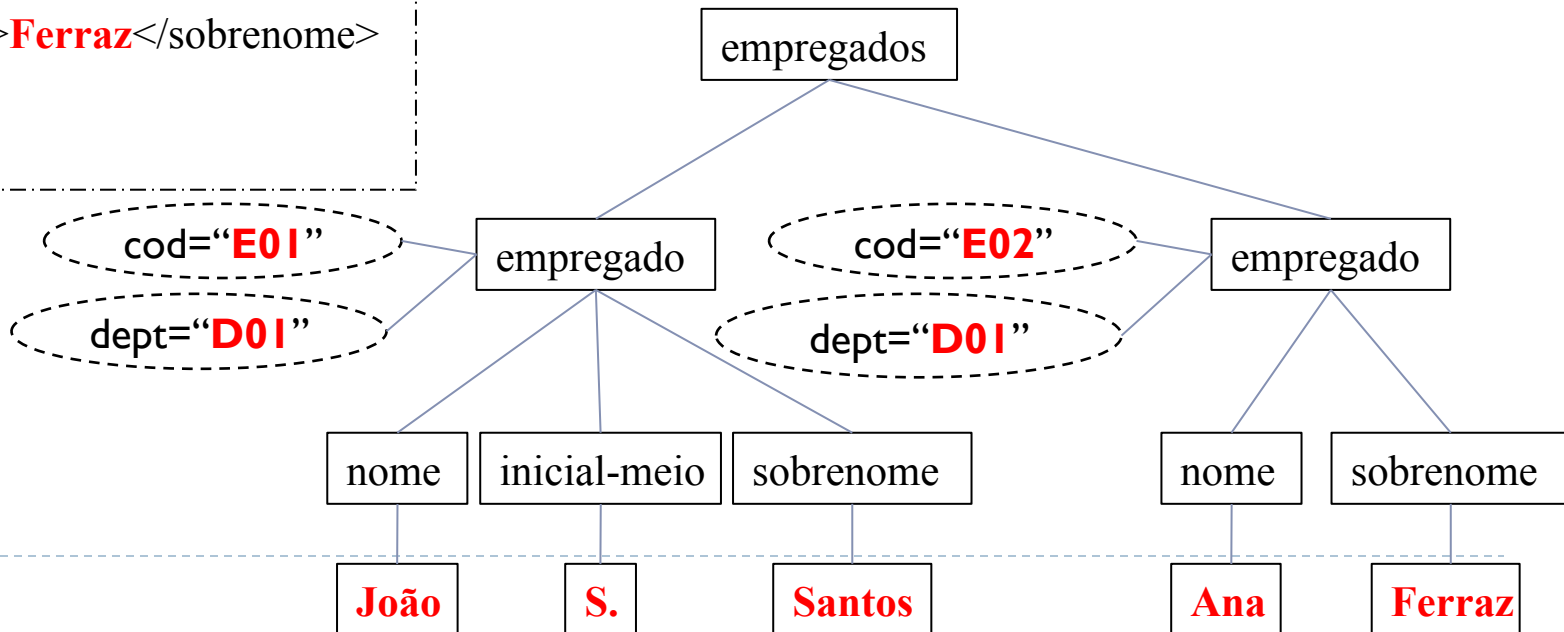
Exemplo de Documento XML

```
<? xml version="1.0" ?>
<empregados>
  <empregado cod="E01" dept="D01">
    <nome>João</nome>
    <inicial-meio>S.</inicial-meio>
    <sobrenome>Santos</sobrenome>
  </empregado>
  <empregado cod="E02" dept="D01">
    <nome>Ana</nome>
    <sobrenome>Ferraz</sobrenome>
  </empregado>
</empregados>
```

Documentos XML como árvores

```
<? xml version="1.0" ?>
<empregados>
  <empregado cod="E01" dept="D01">
    <nome>João</nome>
    <inicial-meio>S.</inicial-meio>
    <sobrenome>Santos</sobrenome>
  </empregado>
  <empregado cod="E02" dept="D01">
    <nome>Ana</nome>
    <sobrenome>Ferraz</sobrenome>
  </empregado>
</empregados>
```

- ▶ Elementos, atributos e **texto** → nodos
- ▶ Relações pai/filho → arestas



Documentos XML Bem-Formados

- ▶ Por representar uma estrutura de árvore, algumas restrições se aplicam a documentos XML
 - ▶ Raiz única (elemento documento ou elemento raiz)
 - ▶ Todas as marcas são fechadas
 - ▶ Elementos são bem aninhados (marcas fecham na ordem inversa à em que foram abertas)
 - ▶ Exemplo de elemento não bem-aninhado
`<empregado><nome>João</empregado></nome>`
 - ▶ Atributos não se repetem no mesmo elemento
 - ▶ Nomes de elementos são sensíveis a maiúsculas e minúsculas

Tipos de Elemento

▶ Composto

- ▶ Contém outros (sub)-elementos

<empregado>

<nome>Ana</nome>

<sobrenome>Ferraz</sobrenome>

</empregado>

▶ Textual

- ▶ Contém somente texto

<nome>Ana**</nome>**

▶ Misto

- ▶ Contém texto e sub-elementos

<endereco> Rua das Flores, 75
<cidade>Rio de Janeiro</cidade>**</endereco>**

▶ Vazio

- ▶ Elemento sem conteúdo

<engenheiro></engenheiro>

<engenheiro/>

Outras considerações importantes

- ▶ Elementos são ordenados
- ▶ Atributos não são ordenados

```
<? xml version="1.0" ?>
<empregados>
  <empregado cod="E01" dept="D01">
    <nome>João</nome>
    <inicial-meio>S.</inicial-meio>
    <sobrenome>Santos</sobrenome>
  </empregado>
  <empregado cod="E02" dept="D01">
    <nome>Ana</nome>
    <sobrenome>Ferraz</sobrenome>
  </empregado>
</empregados>
```

```
<? xml version="1.0" ?>
<empregados>
  <empregado cod="E02" dept="D01">
    <nome>Ana</nome>
    <sobrenome>Ferraz</sobrenome>
  </empregado>
  <empregado cod="E01" dept="D01">
    <nome>João</nome>
    <inicial-meio>S.</inicial-meio>
    <sobrenome>Santos</sobrenome>
  </empregado>
</empregados>
```



Documentos Diferentes

Outras considerações importantes

- ▶ Elementos são ordenados
- ▶ Atributos não são ordenados

```
<? xml version="1.0" ?>
<empregados>
  <empregado dept="D01" cod="E01" >
    <nome>João</nome>
    <inicial-meio>S.</inicial-meio>
    <sobrenome>Santos</sobrenome>
  </empregado>
  <empregado cod="E02" dept="D01">
    <nome>Ana</nome>
    <sobrenome>Ferraz</sobrenome>
  </empregado>
</empregados>
```

```
<? xml version="1.0" ?>
<empregados>
  <empregado cod="E01" dept="D01">
    <nome>João</nome>
    <inicial-meio>S.</inicial-meio>
    <sobrenome>Santos</sobrenome>
  </empregado>
  <empregado cod="E02" dept="D01">
    <nome>Ana</nome>
    <sobrenome>Ferraz</sobrenome>
  </empregado>
</empregados>
```



Elementos x Atributos

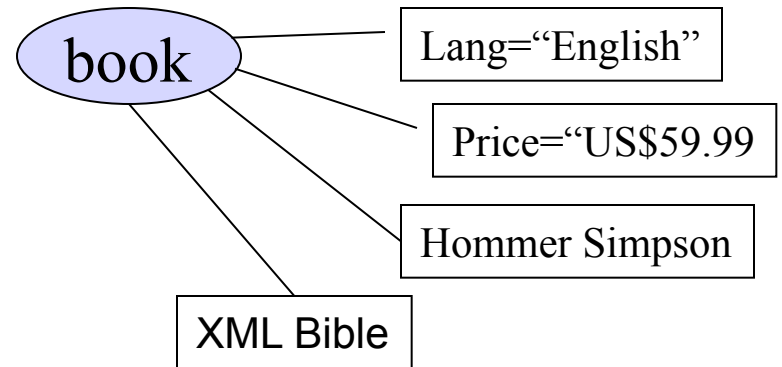
- ▶ Não há regras
- ▶ Atributos apresentam algumas restrições
 - ▶ Não são extensíveis
 - ▶ Não permitem múltiplos valores
 - ▶ Não descrevem estruturas
- ▶ Recomendação: em geral, preferir elementos, e usar atributos para informações secundárias
- ▶ Metadados (dados sobre os dados) devem ser representados como atributos

Ex: `<price currency="US">59.99</price>`

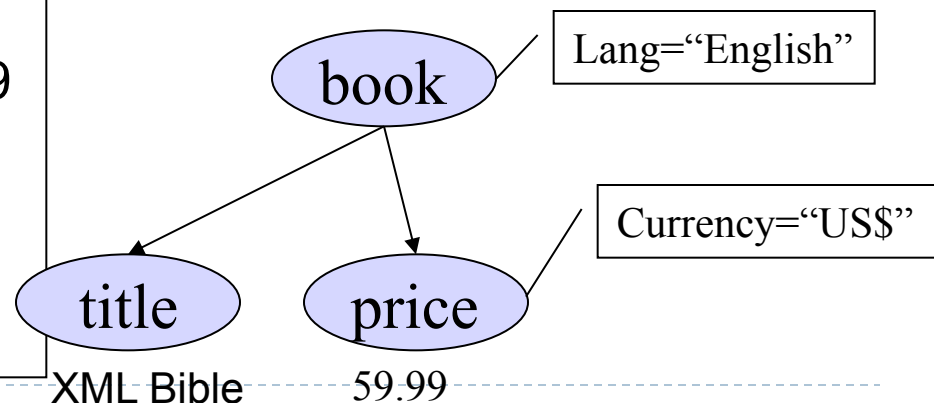


Elementos x Atributos

```
<book lang="English" price="US  
$59.99"  
title="XML Bible"  
author="Hommer Simpson">  
...  
</book>
```



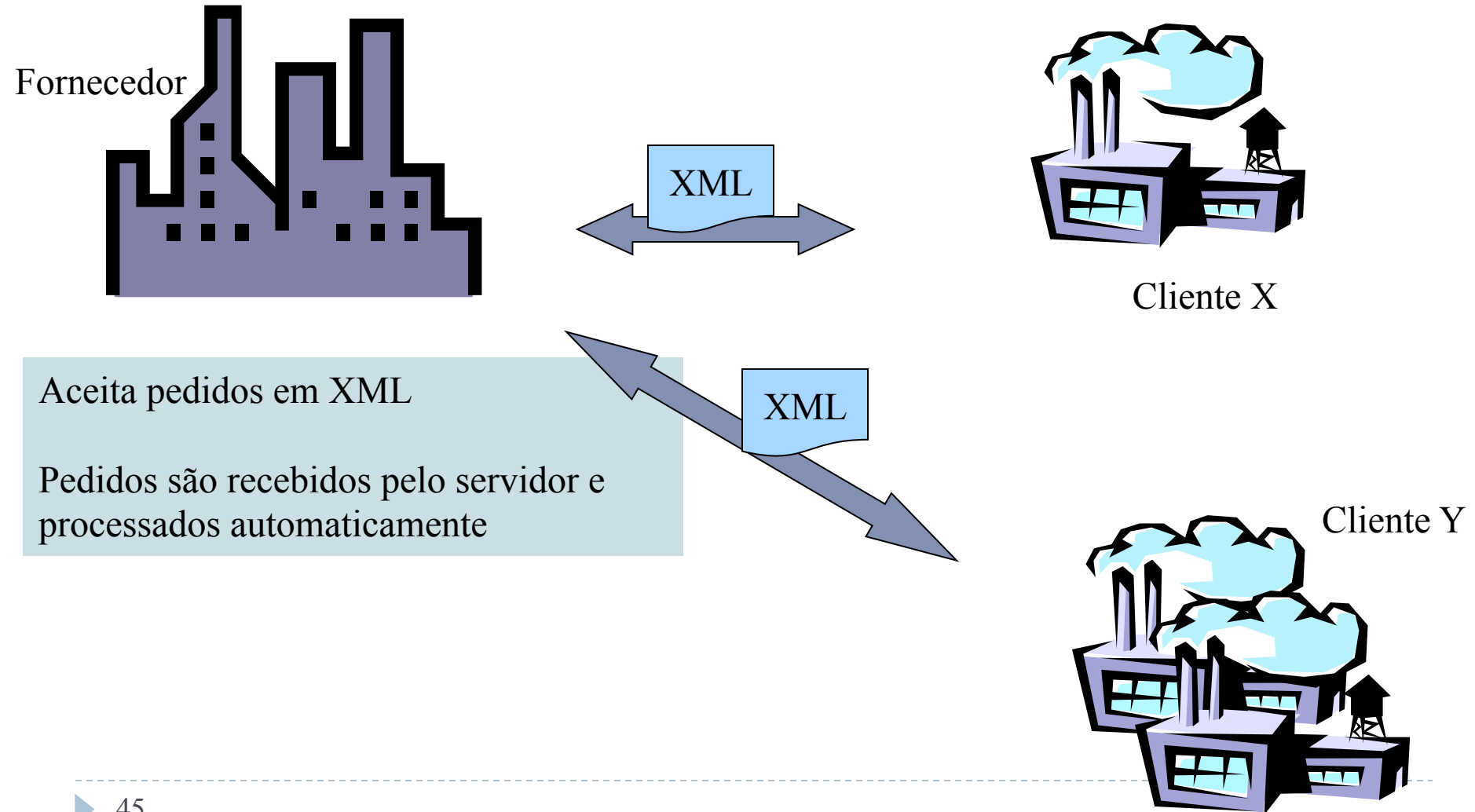
```
<book lang="English">  
  <price currency="US$"> 59.99  
</price>  
  <title>XML Bible </title>  
  ...  
</book>
```



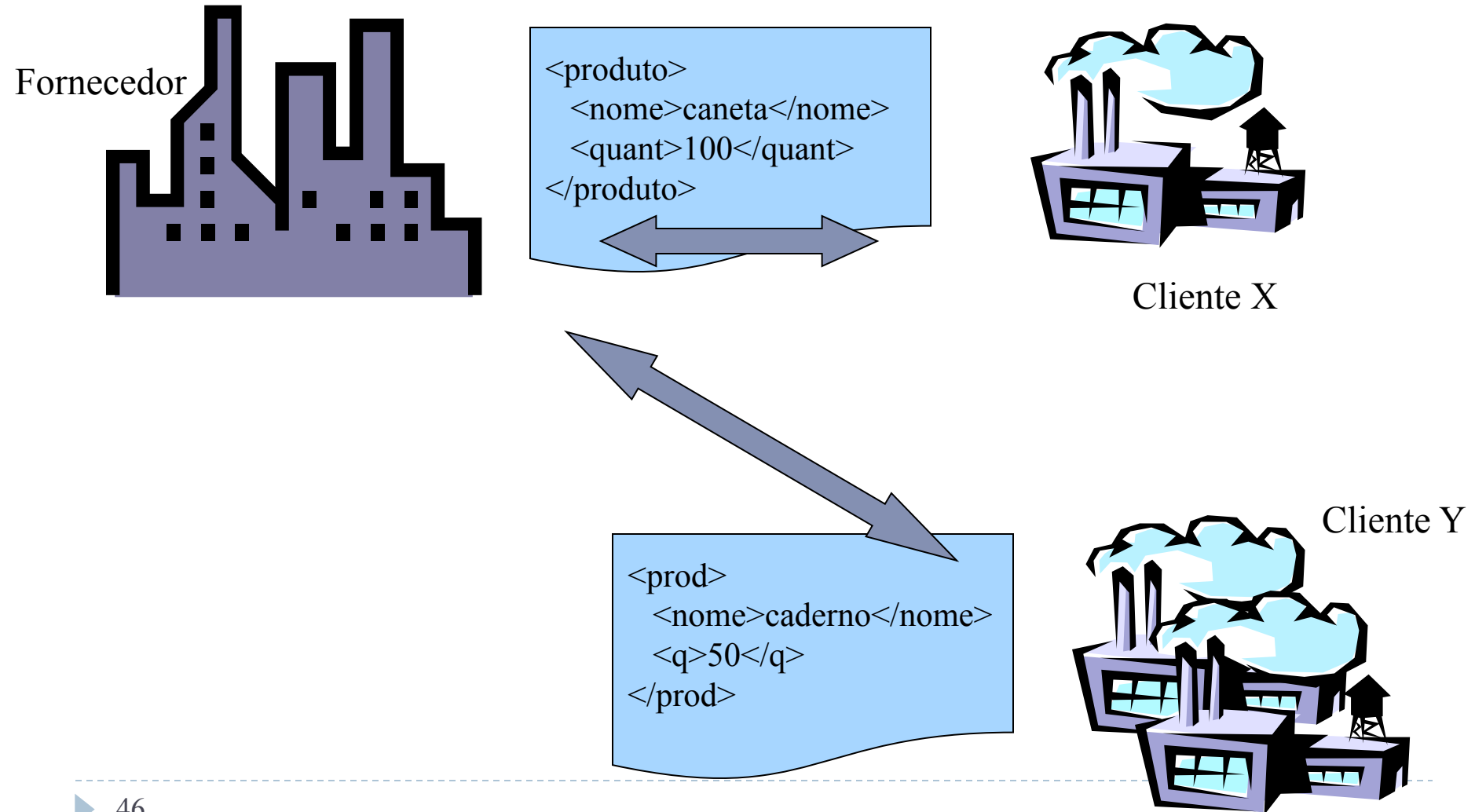
Interoperabilidade

- ▶ Chave do sucesso de XML:
 - ▶ Usuários podem definir suas próprias marcas
- ▶ E a interoperabilidade, como fica?
 - ▶ Vamos considerar um exemplo prático

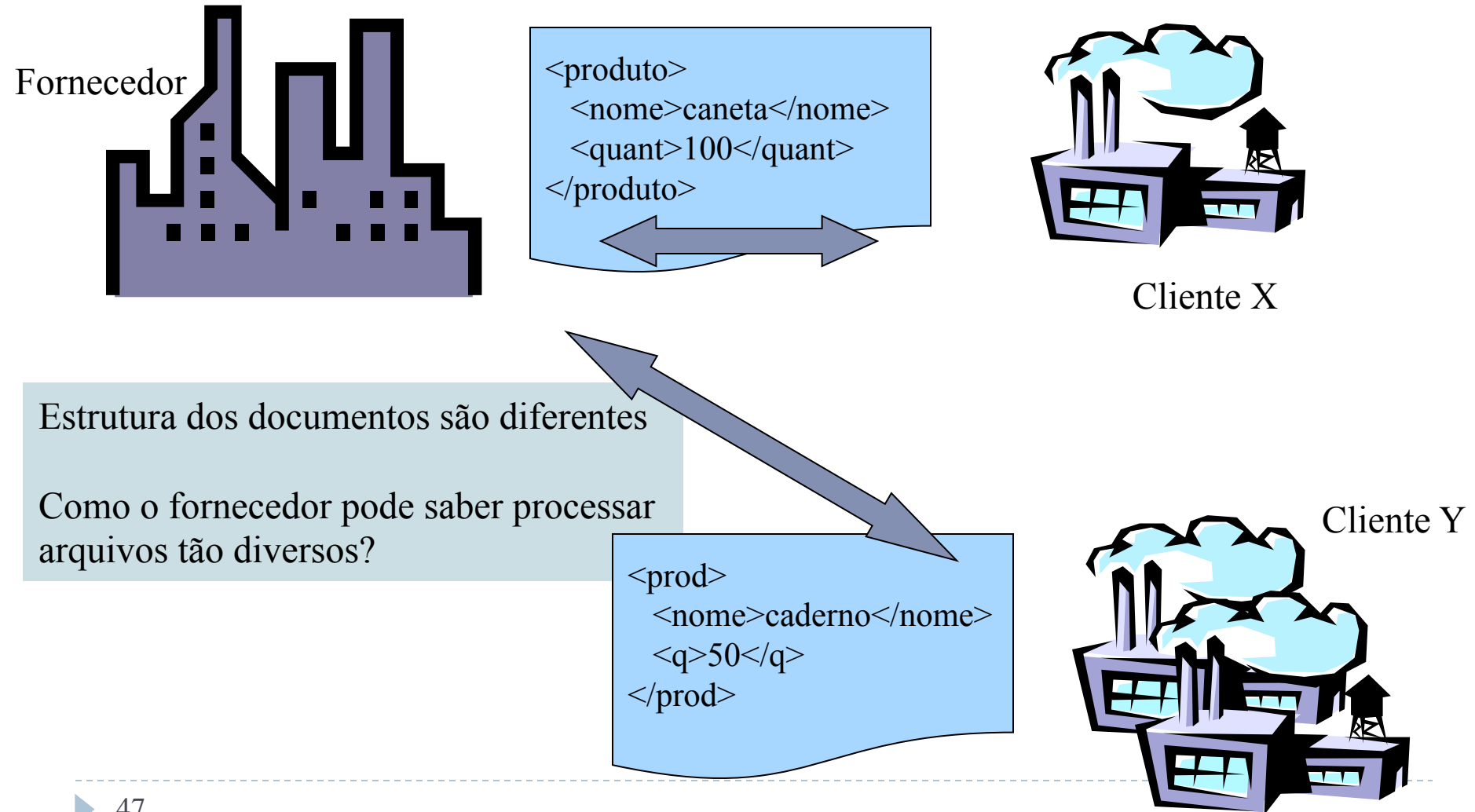
Exemplo prático: interoperabilidade



Exemplo prático: interoperabilidade



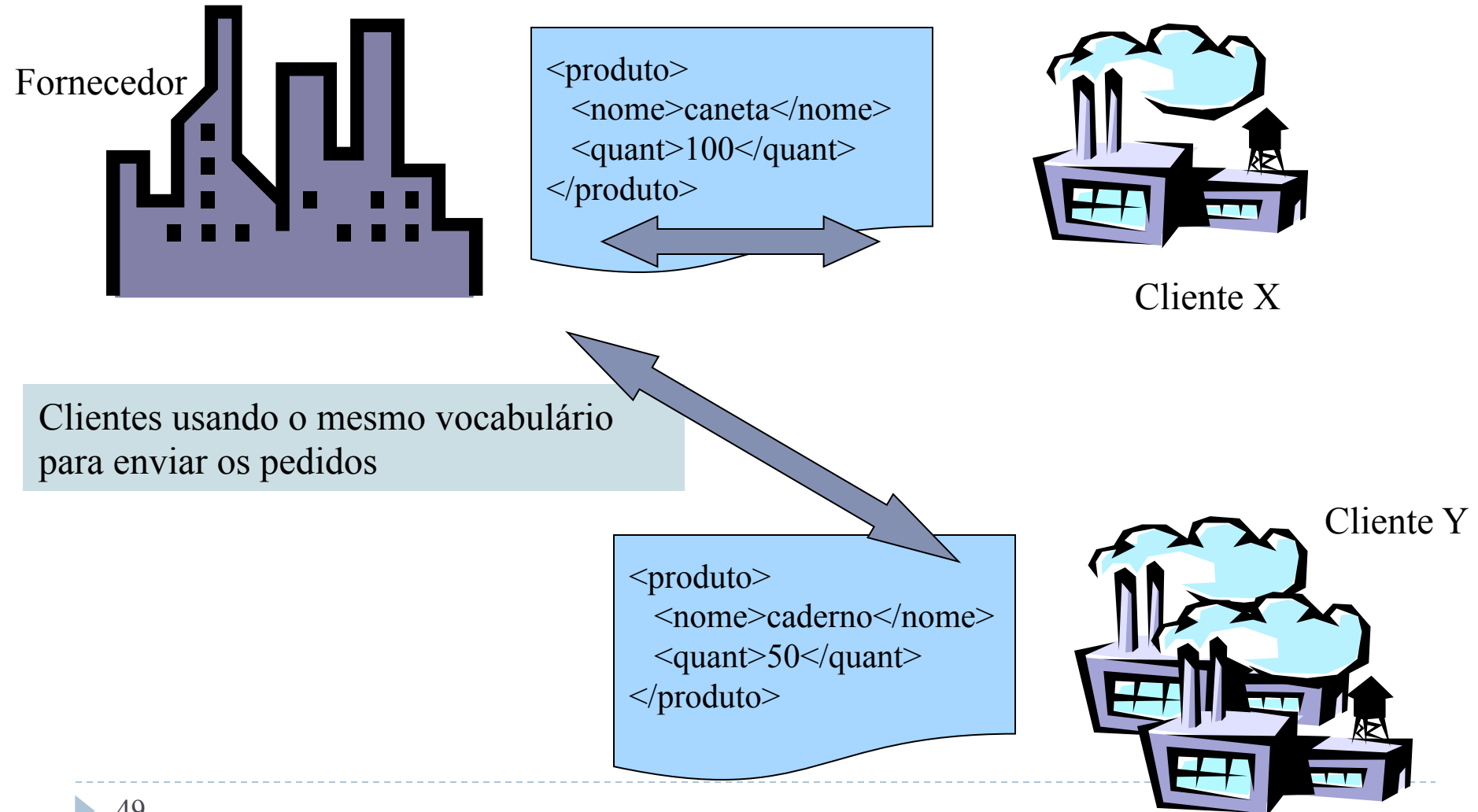
Exemplo prático: interoperabilidade



Solução

- ▶ Pode-se definir um vocabulário usando uma linguagem de esquemas para XML (DTD ou XML Schema)
 - ▶ Fornecedor define o vocabulário (estrutura, nomes das marcas, tipos de dados)
 - ▶ Cliente usa o vocabulário para enviar os pedidos

Interoperabilidade



Exercício

- ▶ Escrever um documento XML para representar uma receita médica
 - ▶ Lembre-se: é importante pensar em como estes documentos serão **estruturados**, e não em como serão **apresentados**
 - ▶ O slide seguinte contém um exemplo de receita médica



Exemplo de Receituário médico



Ana Maria Marina , 7 anos

Uso interno

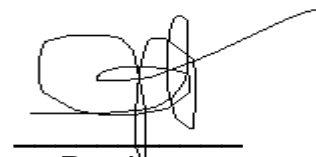
Xarope SemTosse

1 colher 3x ao dia

Uso Externo

Gyellow

aplicar no braço 1x ao dia ao deitar



Dr. Juca

20/10/2001

Vamos testar?

- ▶ Use o Exchanger XML Lite para verificar o documento XML que você criou

ou

- ▶ Use o parser RXP... (baixar do site da disciplina)
 - ▶ `rxp <nome do arquivo XML>`



Exercícios

- ▶ Faça o seu currículo em XML
- ▶ Informações obrigatórias:
 - ▶ Dados pessoais
 - ▶ Formação
 - ▶ Idiomas
 - ▶ Cursos adicionais



Sintaxe em mais detalhes

Versão do XML

- ▶ Existem duas versões da especificação da linguagem XML:
 - ▶ 1.0
 - ▶ 1.1



Histórico da Evolução das versões

- ▶ Versão 1.0: nomes de elementos e atributos baseados no encoding Unicode 2.0
- ▶ Unicode evoluiu para 4.0, e deste modo, os caracteres que não estavam presentes na versão 2.0 não podiam ser usados em nomes de elementos e atributos (hoje já está na versão 5.0)
- ▶ Especificação de nomes na versão 1.0 do XML: tudo que não era explicitamente permitido, era proibido



Versão 1.1

- ▶ Flexibilizou a definição de nomes em XML, mudando a regra do jogo
 - ▶ Tudo que não é explicitamente proibido, é permitido
 - ▶ Isso faz que com a versão de XML não precise mais mudar caso surjam novas versões do Unicode



No curso

- ▶ Aqui no curso ficaremos com a versão 1.0...



Nomes de Elementos e Atributos

- ▶ Não podem começar com:
 - ▶ – (traço)
 - ▶ . (ponto)
 - ▶ dígito



Instrução de Processamento

- ▶ Mecanismo de inserção de informações explícitas em um documento destinadas a alguma aplicação
- ▶ Os parsers XML não interpretam tais informações, assim como não o fazem para comentários; eles simplesmente as repassam para a aplicação
- ▶ Sintaticamente uma instrução de processamento é uma cadeia de caracteres que começa com a configuração `<?>` e termina com `?>`
- ▶ Exemplo

`<para>`

seria bom finalizar esta pagina

`<?ACME-paginator DO:new-page?>` aqui.

`</para>`

Instrução especial XML

- ▶ É uma instrução de processamento especial

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
```

- ▶ **Parâmetros**

- ▶ **version** indica a versão da linguagem (1.0 ou 1.1) - obrigatório
- ▶ **encoding** indica a codificação de caracteres utilizada no documento - opcional



Instrução especial XML

▶ Parâmetros (cont.)

▶ **standalone** - opcional

- ▶ ="yes", indica que não existem declarações externas que afetam a interpretação (default)
- ▶ ="no", indica que um conjunto de declarações definido externamente contém informação que afeta a interpretação do conteúdo do documento.
- ▶ O valor "no" deve ser usado se qualquer elemento, atributo ou entidade externa for definida em uma DTD externa



Encoding

- ▶ Utilizar atributo encoding na declaração XML do prólogo

ex: `<?xml version="1.0" encoding="ISO-8859-1"?>`

- ▶ Pode-se utilizar declaração de encoding como parte de uma instrução de processamento separada, após a declaração XML, mas antes do caractere aparecer

ex: `<?xml encoding="ISO-8859-1" ?>`



Conjuntos de caracteres

- ▶ UTF-8
 - ▶ UTF-16
 - ▶ ASCII
- } Unicode
- ▶ 1 byte, 7 bits --> 128 combinações
 - ▶ ISO 8859-1 Latin-1
 - ▶ 1 byte, 8 bits --> 256 combinações (ASCII + caracteres para maioria das línguas da Europa Ocidental - inclusive Português)
 - ▶ ISO 8859-(2...15)
 - ▶ 1 byte, 8 bits --> 256 combinações (ASCII + caracteres para outros conjuntos de línguas)



Comentários

- ▶ Começam com `<! --` e terminam com `-->`
- ▶ Todo dado entre essas marcas é ignorado pelo processador XML
- ▶ Não podem acontecer antes da instrução de processamento (declaração XML) que deve ser a primeira sentença de um documento, nem dentro de alguma marcação (tag inicial, tag final, instrução de processamento, etc.)
- ▶ A sequência `--` não pode aparecer em um comentário, exceto como parte dos delimitadores do comentário



Seções CDATA

- ▶ Normalmente o texto que aparece entre os delimitadores `<` e `>` são considerados marcações. Exceção é feita aos textos entre delimitadores de seção CDATA, que são considerados caracteres de dado
- ▶ Os delimitadores de abertura e fechamento da seção são, respectivamente, `<![CDATA[` e `]]>`
- ▶ A única sequência de caracteres que não pode aparecer em uma seção CDATA é `]]>`



Declaração CDATA

- ▶ As seções CDATA são úteis quando se deseja que todos os caracteres de um texto sejam interpretados como caracteres e não como elementos de marcação
- ▶ Exemplos são textos contendo os caracteres <, >, &, etc., comuns em trechos de código de programas.

- ▶ Exemplo:

```
<![CDATA[ Em XML a entidade &lt; eh built- in ]]>
```

- ▶ Resultado depois do documento ser processado:

```
"Em XML a entidade &lt; eh built- in".
```





Revisando...



```

<?xml version="1.0" encoding="ISO-8859-1" ?>
<!-- ?xml-stylesheet type="text/css" href="aula.css"? -->
<!-- Apresenta-se o texto, algoritmo e código para resolver o problema de escolher
o menor entre tres numeros. Conclui-se com dois testes -->
- <aulaml id="maior01" prof="Graça" titulo="Verificar o maior entre 3 números">
  <curso cod="sce-180" nome="Introdução à Ciência da Computação" />
  <index termo="maior" />
+ <quadro id="q1" tipo="teoria">
- <quadro id="q2" tipo="teoria">
  - <texto>
    <paragrafo>Encontra o maior número entre A, B e C</paragrafo>
  - <sequencia>
    <passo>se A > B entao MAIOR = A senao MAIOR = B</passo>
    <passo>se MAIOR < C entao MAIOR = C</passo>
    <passo>fim</passo>
  </sequencia>
</texto>
- <codigo>
  <![CDATA[      if A > B then MAIOR = A else MAIOR = B;      ]]>
</codigo>
+ <teste>
</quadro>
</aulaml>

```

```

<?xml version="1.0" encoding="ISO-8859-1" ?>
<!-- ?xml stylesheet type="text/css" href="aula.css"? -->
<!-- Apresenta algoritmo e código para resolver o problema de escolher
o menor

```

Instruções de Processamento: Mecanismo de inserção de informações explícitas em um documento que são destinadas a alguma aplicação. Começa com <? e termina com ?>

```

- <aula
  <cu
  <ind
+ <qu
- <qu
- <te
  <paragrafo> Encontra o maior número entre A, B e C</paragrafo>
- <sequencia>
  <passo> se A > B entao MAIOR = A senao MAIOR = B</passo>
  <passo> se MAIOR < C entao MAIOR = C</passo>
  <passo> fim</passo>
</sequencia>
</texto>
- <codigo>
  <![CDATA[      if A > B then MAIOR = A else MAIOR = B;      ]]>
</codigo>
+ <teste>
</quadro>
</aulaml>

```

```
C:\Gracacursos\jai2000\curitiba aula-Maior.xml - AT&T Internet Explorer
File Edit View Favorites Tools Help

<?xml version="1.0" encoding="ISO-8859-1" ?>
<!-- ?xml-stylesheet type="text/css" href="aula.css"? -->
<!-- Apresenta-se o texto, algoritmo e código para resolver o problema de escolher
o menor entre tres numeros. Conclui-se com dois testes -->
- <aulaml="maior01" prof="Graça" titulo="Verificar o maior entre 3 números">
  <curso="180" nome="Introdução à Ciência da Computação" />
  <index="1" />
  <sequencia>
    </sequencia>
  </texto>
  - <codigo>
    <![CDATA[ if A > B then MAIOR = A else MAIOR = B; ]]>
  </codigo>
  <teste>
  </quadro>
</aulaml>
```

Comentários começam com <!-- e terminam com --> e são ignorados. Não podem acontecer antes da instrução de declaração XML nem dentro de marcações; não podem conter a seqüência --

Referências a Entidades são marcações que são substituídas com caracteres de dados no processamento do documento.

As cinco entidades a seguir são predefinidas por XML:

`&`; `<`; `>`; `"`; `'`;

```

<?xml v
<!-- ?x
<!-- Ap
o menor
- <aulaml
  <curso
  <index
+ <quadro id= q1 tip
- <quadro id="q2" tip
- <texto>
  <paragrafo>Enc...o maior número entre A, B e C</paragrafo>
- <sequencia>
  <passo>se A > B entao MAIOR = A senao MAIOR = B</passo>
  <passo>se MAIOR < C entao MAIOR = C</passo>
  <passo>fim</passo>
</sequencia>
</texto>
- <codigo>
  <![CDATA[
    if A > B then MAIOR = A else MAIOR = B;
  ]]>
</codigo>
+ <teste>
</quadro>
</aulaml>
  
```



```

<?xml version="1.0" encoding="ISO-8859-1" ?>
<!-- ?xml-stylesheet type="text/css" href="aula.css"? -->
<!-- Apresenta-se o texto, algoritmo e código para resolver o problema de escolher
o menor entre tres numeros. Conclui-se com dois testes -->
- <aulaml id="maior01" prof="Graça" titulo="Verificar o maior entre 3 números">
  <curso cod="sce-180" nome="Introdução à Ciência da Computação" />
  <index termo="maior" />
+ <quadro id="q1" tipo="teoria">
- <quadro id="q2" tipo="teoria">
  - <texto>
    <paragrafo>Encontra o maior número entre A, B e C</paragrafo>
  - <sequencia>
    <passo>se A > B entao MAIOR = A senao MAIOR = B</passo>
    <passo>se MAIOR < C entao MAIOR = C</passo>
    <passo>fim</passo>
  </sequencia>
</texto>
  - <codigo>

```

<passo>se A > B

entao MAIOR = A senao MAIOR = B

</passo>

<passo>se MAIOR < C entao MAIOR = C

</passo>

```

<?xml version="1.0" encoding="ISO-8859-1" ?>
<!-- ?xml-stylesheet type="text/css" href="aula.css"? -->
<!-- Apresenta-se o texto, algoritmo e código para resolver o problema de escolher
o menor entre tres numeros. Conclui-se com dois testes -->
- <aulaml id="maior01" prof="Graça" titulo="Verificar o maior entre 3 números">
  <curso cod="sce-180" nome="Introdução à Ciência da Computação" />
  <index termo="maior" />
+ <quadro id="q1" tipo="teoria">
- <quadro id="q2" tipo="teoria">
  - <texto>
    <paragrafo>Encontra o maior número entre A, B e C</paragrafo>
  - <sequencia>
    <passo>se A > B entao MAIOR = A senao MAIOR = B</passo>
    <passo>se MAIOR < C entao MAIOR = C</passo>
    <passo>fim</passo>
  </sequencia>
</texto>
- <codigo>
  <![CDATA[          if A > B then MAIOR = A else MAIOR = B;          ]]>
</codigo>
+ <teste>
</quadro>
</aulaml>

```

CDATA: todo o texto que aparece entre delimitadores de seção CDATA que são considerados caracteres de dado:

`<![CDATA[...]]>`

Exercício

1. Escreva um documento XML que tenha “5 < 4” como o valor de um elemento
2. Escreva um documento XML que tenha uma entidade de texto >
3. Agora faça a entidade aparecer no resultado assim como foi escrita no documento XML

- ▶ Basta abrir o documento no browser (Firefox ou IE) para ver o resultado, ou então
- ▶ Use o RXP para ver o resultado em cada exercício, com os parâmetros -bm, assim:

```
rxp -bm <arquivo XML>
```

Exercício

- ▶ Escreva um documento XML simples, e teste se ele é um documento bem-formatado. Revise as características de documentos bem-formatados e teste todas elas. Veja o tipo de erro gerado pelo parser em cada um deles.



Documentos XML *bem formados*

- ▶ Tipagem fraca
 - ▶ É apenas um parser em uma árvore rotulada
 - ▶ Não verifica a estrutura em si:
 - ▶ a hierarquia dos elementos e seus atributos
 - ▶ Não se pode garantir por exemplo:
 - ▶ Toda instância de livro deve ter um elemento “preço” que inclui um atributo “moeda”

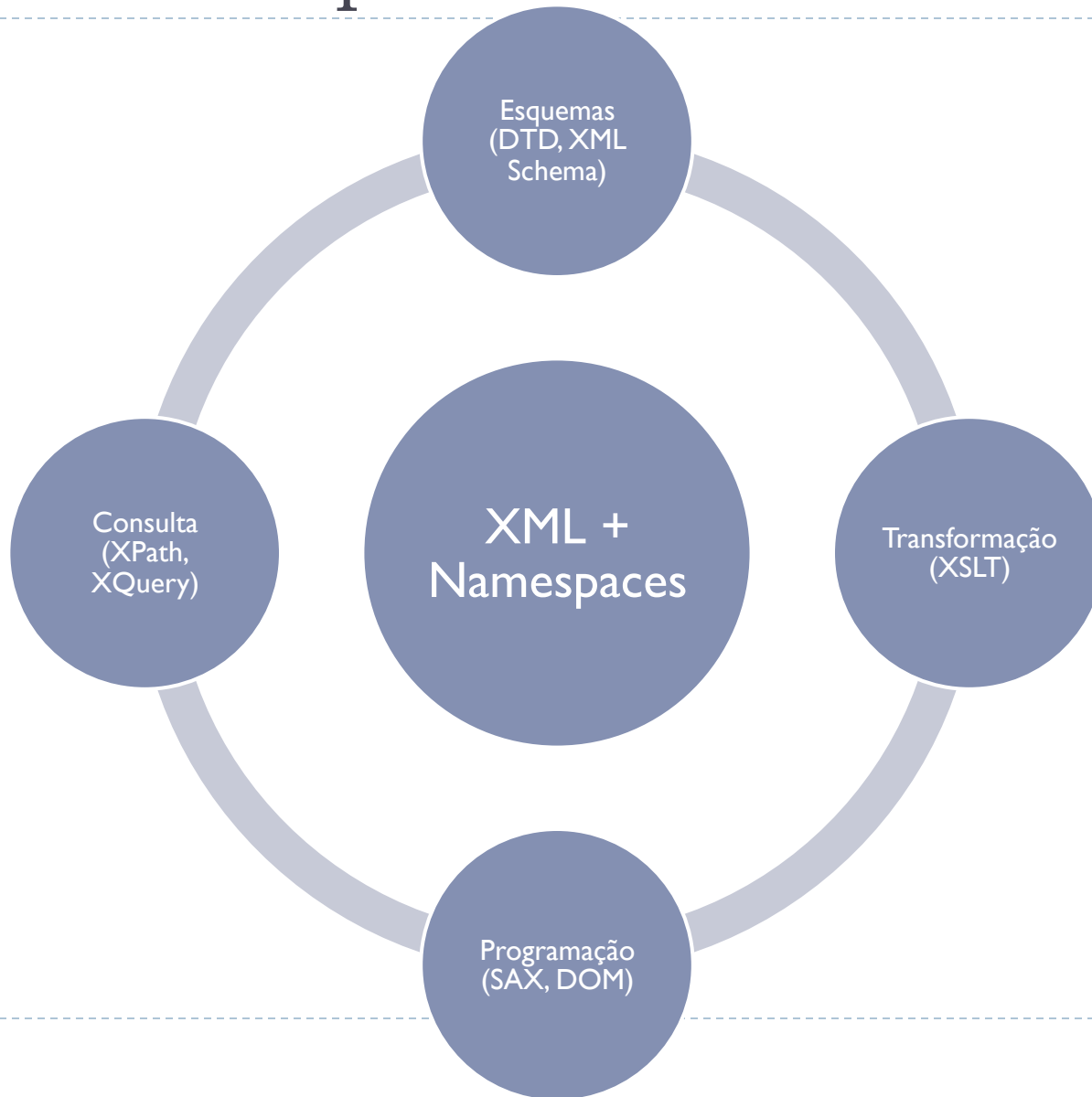


Documentos XML *bem formados*

- ▶ Tecnicamente **não precisam** começar com a declaração XML, mas o W3C recomenda
 - ▶ `<?xml version = “1.0”?>`
 - ▶ Instrução de processamento que indica que o documento está escrito em XML em uma determinada versão



XML e Namespaces: Núcleo





Namespaces

Namespaces (<http://www.w3.org/TR/xml-names>)

- ▶ Usual: vocabulários já definidos são usados para construir novos vocabulários (reuso)
- ▶ O que acontece se dois vocabulários que estão sendo reusados possuem nomes de marcas iguais, mas em contextos diferentes?
- ▶ Como diferenciar qual marca veio de onde?

Exemplo

- ▶ Vocabulário da matemática:
 - ▶ Marcas: conjunto, **elemento**, ...
- ▶ Vocabulário da química:
 - ▶ Marcas: **elemento** , ...
- ▶ Vocabulário a ser criado: conceitos do ensino médio
 - ▶ Serão utilizados os vocabulários da matemática e da química, entre outros

Problema

- ▶ Como distinguir um **elemento** de um conjunto da matemática, de um **elemento** químico?
- ▶ Solução: uso de namespaces
- ▶ A cada vocabulário é associado um *namespace*, identificado por uma URI

Exemplo

- ▶ Namespace da matemática: <http://www.matematica.com>
- ▶ Namespace da química: <http://www.quimica.com>
- ▶ URIs são usadas por serem identificadores únicos
 - ▶ Não é necessário que o endereço exista na Web

Exemplo

- ▶ Ao referenciar um elemento, usa-se o namespace para fazer a desambiguação
- ▶ Para encurtar, usa-se um prefixo para referenciar o namespace

Exemplo – documento de ensino

```
<ensino xmlns:m="http://www.matematica.com"
        xmlns:q="http://www.quimica.com">
  <m:conjunto>
    <m:elemento>I</m:elemento>
    <m:elemento>3</m:elemento>
  </m:conjunto>
  <q:elemento>Ca</q:elemento>
</ensino>
```

Exemplo – documento de ensino

```
<ensino xmlns:m="http://www.matematica.com"
        xmlns:q="http://www.quimica.com">
```

```
<m:conjunto>
```

```
<m:elemento>1</m:elemento>
```

```
<m:elemento>3</m:elemento>
```

```
</m:conjunto>
```

```
<q:elemento>Ca</q:elemento>
```

```
</ensino>
```

**Declaração dos namespaces
(atributo xmlns)**

Exemplo – documento de ensino

```
<ensino xmlns:m="http://www.matematica.com"
        xmlns:q="http://www.quimica.com">
  <m:conjunto>      Prefixos dos namespaces
    <m:elemento> I </m:elemento>
    <m:elemento> 3 </m:elemento>
  </m:conjunto>
  <q:elemento>Ca</q:elemento>
</ensino>
```


Exemplo – documento de ensino

```
<ensino xmlns:m="http://www.matematica.com"  
        xmlns:q="http://www.quimica.com">
```

```
<m:conjunto>
```

Elementos da matemática

```
<m:elemento>1</m:elemento>
```

```
<m:elemento>3</m:elemento>
```

```
</m:conjunto>
```

```
<q:elemento>Ca</q:elemento>
```

```
</ensino>
```

Exemplo – documento de ensino

```
<ensino xmlns:m="http://www.matematica.com"
        xmlns:q="http://www.quimica.com">
  <m:conjunto>
    <m:elemento>I</m:elemento>
    <m:elemento>3</m:elemento>
  </m:conjunto>
  <q:elemento>Ca</q:elemento>
</ensino>
```

Elemento da química

Múltiplos Namespaces

```
<?xml version="1.0"?>
<!-- both namespace prefixes are available throughout -->
<bk:book xmlns:bk='urn:loc.gov:books'
          xmlns:isbn='urn:ISBN:0-395-36341-6'>
  <bk:title>Cheaper by the Dozen</bk:title>
  <isbn:number>1568491379</isbn:number>
</bk:book>
```



Outro Exemplo

```
<aaa >  
  <bbb >  
    <ccc />  
  </bbb>  
<BBB >  
  <CCC />  
</BBB>  
<x111 >  
  <x222 />  
</x111>  
</aaa>
```



Declarações em todos os elementos

```
<lower:aaa xmlns:lower = "http://etc.org/lowercase" >
  <lower:bbb xmlns:lower = "http://etc.org/lowercase" >
    <lower:ccc xmlns:lower = "http://etc.org/lowercase" />
  </lower:bbb>
  <upper:BBB xmlns:upper = "http://etc.org/uppercase" >
    <upper:CCC xmlns:upper = "http://etc.org/uppercase" />
  </upper:BBB>
  <xnumber:x111 xmlns:xnumber = "http://etc.org/xnumber" >
    <xnumber:x222 xmlns:xnumber = "http://etc.org/xnumber" />
  </xnumber:x111>
</lower:aaa>
```

▶ Vamos testar com o RXP?

rxp -N <arquivo XML>

▶ Testem inserir erros de namespace...



Declarações só em alguns elementos

```
<lower:aaa xmlns:lower = "http://zvon.org/lowercase" >
  <lower:bbb >
    <lower:ccc />
  </lower:bbb>
  <upper:BBB xmlns:upper = "http://zvon.org/uppercase" >
    <upper:CCC />
  </upper:BBB>
  <xnumber:x111 xmlns:xnumber = "http://zvon.org/xnumber" >
    <xnumber:x222 />
  </xnumber:x111>
</lower:aaa>
```



Declaração na raiz

```
<lower:aaa xmlns:lower = "http://etc.org/lowercase"
  xmlns:upper = "http://etc.org/uppercase"
  xmlns:xnumber = "http://etc.org/xnumber" >
  <lower:bbb >
    <lower:ccc />
  </lower:bbb>
  <upper:BBB >
    <upper:CCC />
  </upper:BBB>
  <xnumber:x111 >
    <xnumber:x222 />
  </xnumber:x111>
</lower:aaa>
```



Namespace Default

- ▶ Namespaces não têm que ser declarados explicitamente com prefixos
- ▶ O atributo xmlns define o namespace default que é usado para o elemento onde ele ocorre e para seus filhos e descendentes

```
<aaa >
  <bbb xmlns = "http://etc.org/lowercase" >
    <ccc />
  </bbb>
  <BBB xmlns = "http://etc.org/uppercase" >
    <CCC />
  </BBB>
  <x111 xmlns = "http://etc.org/xnumber" >
    <x222 />
  </x111>
</aaa>
```


Exemplo

- ▶ Elementos podem até pertencer a diferentes namespaces embora eles tenham os mesmos prefixos!

```
<aaa >
  <lower:bbb xmlns:lower = "http://etc.org/lowercase" >
    <lower:ccc />
  </lower:bbb>
  <lower:BBB xmlns:lower = "http://etc.org/uppercase" >
    <lower:CCC />
  </lower:BBB>
  <lower:x111 xmlns:lower = "http://etc.org/xnumber" >
    <lower:x222 />
  </lower:x111>
</aaa>
```

☹ ***Mas evite isso, para não confundir!***

Namespaces e seu uso

- ▶ Namespaces são amplamente usados nas diversas iniciativas associadas ao XML

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
```

```
<xsl:template match="pattern">
```

```
<produto><xsl:value-of select="."/></produto>
```

```
</xsl:template>
```

```
</xsl:stylesheet>
```



Agora que já sabemos o que é
XML...

Porque XML?



Extensibilidade e estrutura

- ▶ Em XML, um autor ou uma comunidade de autores inventam livremente as tags que lhes pareçam úteis para marcar os componentes de um documento
- ▶ Exemplo: diversas formas de representar uma data

```
<date> 5 janeiro 2000 </date>
```

```
<date>
```

```
  <ano> 2000 </ano>
```

```
  <mes> 01 </mes>
```

```
  <dia> 05 </dia>
```

```
</date>
```

```
<date format='ISO-8601'> 2000-01-05 </date>
```

- ▶ Grande liberdade de escolha das estruturas de dados facilita a **troca de dados**



Interoperabilidade

- ▶ Todos os dados podem ser vistos como documentos XML e não mais como arquivos no formato X ou Y
- ▶ Consequências:
 - ▶ Um servidor de documentos XML é suscetível de responder a um conjunto de necessidades de uma organização
 - ▶ Um simples editor de textos pode tratar o conjunto de dados de uma organização
 - ▶ A interoperabilidade dos utilitários está assegurada



Modularidade e reutilização

- ▶ Cada usuário é livre para definir suas próprias estruturas de documento
- ▶ Ele pode também estar conforme as estruturas tipadas, chamadas DTD
- ▶ Cada comunidade pode propor as estruturas normalizadas
- ▶ A validação a um DTD permite a automatização no tratamento dos dados e assegura uma possibilidade de controle de integridade



Acesso à fontes de informação heterogêneas

- ▶ A consulta e troca de dados entre as base de dados heterogêneas é complexa
- ▶ XML contribui para minimizar este problema: formato de troca normalizado, genérico, independente de plataforma



Acesso à fontes de informação heterogêneas

- ▶ A indexação e consulta de bases de documentos pode se beneficiar de informações estruturais e textuais
 - ▶ Pesquisa por palavras-chaves: Jorge+Amado retorna todos os documentos contendo as palavras Jorge e Amado
 - ▶ Pesquisa estrutural: pesquisa os documentos cujo autor é Jorge Amado (ie. os documentos contendo um elemento autor, ou escrito-por contendo Jorge e Amado)



XML no Mercado...

- ▶ **Descrever documentos armazenados**
 - ▶ Memória intra e inter organizacional
- ▶ **Descrever dados armazenados**
 - ▶ SGBDs nativos, habilitados e híbridos
- ▶ **Intercambiar dados**
 - ▶ Integrar sistemas
 - ▶ Aplicações B2B
- ▶ **Criar novas linguagens**
 - ▶ WML (Wap Markup Language)



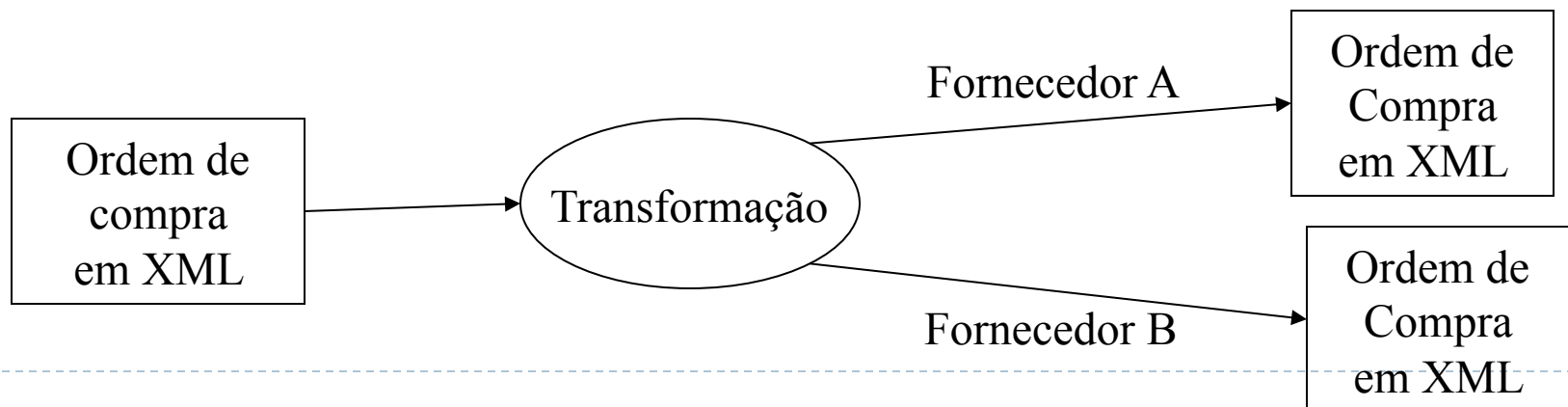
XML no Mercado...

- ▶ **Bancos de Dados e Aplicações**
 - ▶ Microsoft Office 2000, Oracle, Sybase
- ▶ **Metadados:**
 - ▶ OMG-MOF XMI proposal to W3C (IBM, Unisys, Oracle, Rational, Platinum, Sybase ...)
- ▶ **Desenvolvedores de ferramentas:**
 - ▶ Java SAX, Java DOM, etc.
- ▶ **Grupos de padronização:**
 - ▶ ISO 11179, ANSI X3L8, Dublin Core, EDI ...



XML no Mercado

- ▶ XML está se tornando uma plataforma padrão para os processos entre empresas dos quais depende o comércio eletrônico B2B. – W. Lewis
- ▶ B2B e-commerce: empresas que centralizam múltiplos vendedores e compradores, compatibilizando ordens de compra e venda entre eles.



XML

Apenas uma linguagem de marcação?

- ▶ A linguagem XML tem associada uma série de iniciativas:
 - ▶ XSL
 - ▶ SAX, DOM
 - ▶ DTD, XML Schema
 - ▶ XLink and XPointer
 - ▶ XPath, XQuery
 - ▶ RDF, OWL
 - ▶ Serviços Web (WSDL, etc.)
 - ▶ etc.



Durante o curso ...

- ▶ Iremos ver algumas dessas iniciativas
- ▶ Na aula que vem, começamos com DTD

