Sumário

1 Aplicações Não-Convencionais

2 Revisão e Dicas de Modelagem Conceitual

3 BD Orientado a Objetos e Objeto-Relacional

4 BD Temporal

5 BD Geográfico

6 BD XML

XML (*eXtensible Markup Language*)

- Tecnologia desenvolvida pela W3C
  - W3C: *World Wide Web Consortium*
    - definição de padrões para a *Web*
    - consórcio formado pela academia e indústria
- Padrão para representação e transferência de dados
- Motivações para utilização crescente da XML
  - aplicações sobre a *Web*
    - extração, manipulação, integração, transferência e publicação de dados
  - aplicações que lidam com dados de natureza semi-estruturada
    - exemplos: anúncios classificados, livros, ...
    - preferência por docs XML ao invés de dados em BDs relacionais

Protocolos XML

- Definidos em diversos domínios e tecnologias
  - comércio eletrônico
    - CXML, eBisXML, ...
  - referências bibliográficas
    - DBLP, SIGMOD, ...
  - sistemas de informação geográfica
    - SVG, GML, ...
  - dispositivos móveis
    - WAP, WML, ...
  - *web services*
    - SOAP, WSDL, ...
  - ...

Uso Extensivo de Protocolos XML...

- Necessidades
  - métodos de acesso a dados XML pelos programas de aplicação
  - projeto da estrutura dos dados XML
  - facilidades para armazenamento e manipulação de dados XML persistentes
  - ...
- A tecnologia de Banco de Dados (BD) é útil neste contexto

Tecnologia XML x Tecnologia BD

- Similaridades
  - documentos XML mantém coleções de dados
  - tecnologia XML oferece mecanismos para definição e manipulação de dados
    - DTD e XSD, XPath e XQuery, DOM, ...
- Diferenças
  - dado XML não é um dado convencional
    - **dado semi-estruturado**
  - tecnologia XML é carente de alguns mecanismos de gerenciamento de dados
    - integridade, visões, gerência adequada transações, ...

Dado Semi-Estruturado

- Principais características
  - estrutura heterogênea
  - estrutura auto-descritiva
  - estrutura parcial

Estrutura heterogênea

- Cada instância de dado pode ter um esquema particular

```
<autor>
    <nome>Joao Silva</nome>
    <endereco>rua B,23</endereco>
    <eMail>jsilva@inf.ufsc.br</eMail>
</autor>
```

```
<autor>
    <nome>Ana Ramos</nome>
    <endereco>
        <rua>Brasil</rua>
        <numero>767</numero>
        <cidade>Fpolis</cidade>
    </endereco>
    <fone>33313333</fone>
    <fone>33313332</fone>
</autor>
```

Estrutura auto-descritiva

- Cada instância de dado carrega o seu esquema

```
<autor>
    <nome>Ana Ramos</nome>
    <endereco>
        <rua>Brasil</rua>
        <numero>767</numero>
        <cidade>Fpolis</cidade>
    </endereco>
    <fone>33313333</fone>
    <fone>33313332</fone>
</autor>
```

Estrutura parcial

- Apenas parte da descrição de uma instância pode ser estruturada

```
<capítulo numero = 2 titulo = "Tecnologia XML">
   Este capítulo descreve ... XML<ref>(Mel03)</ref>.
XML é um padrão ...
     <secao numero = 1>
          <titulo>DTD</titulo>
          Esta seção descreve ...
     </secao>
     ...
</capítulo>
```

XML & BD

- Dados XML não são naturalmente adequados para armazenamento em BDs

Dado de BD	Dado XML
representação homogênea	representação heterogênea
esquema independente dos dados	representação auto-descritiva
totalmente estruturado	estrutura parcial
esquema enxuto	esquema extenso

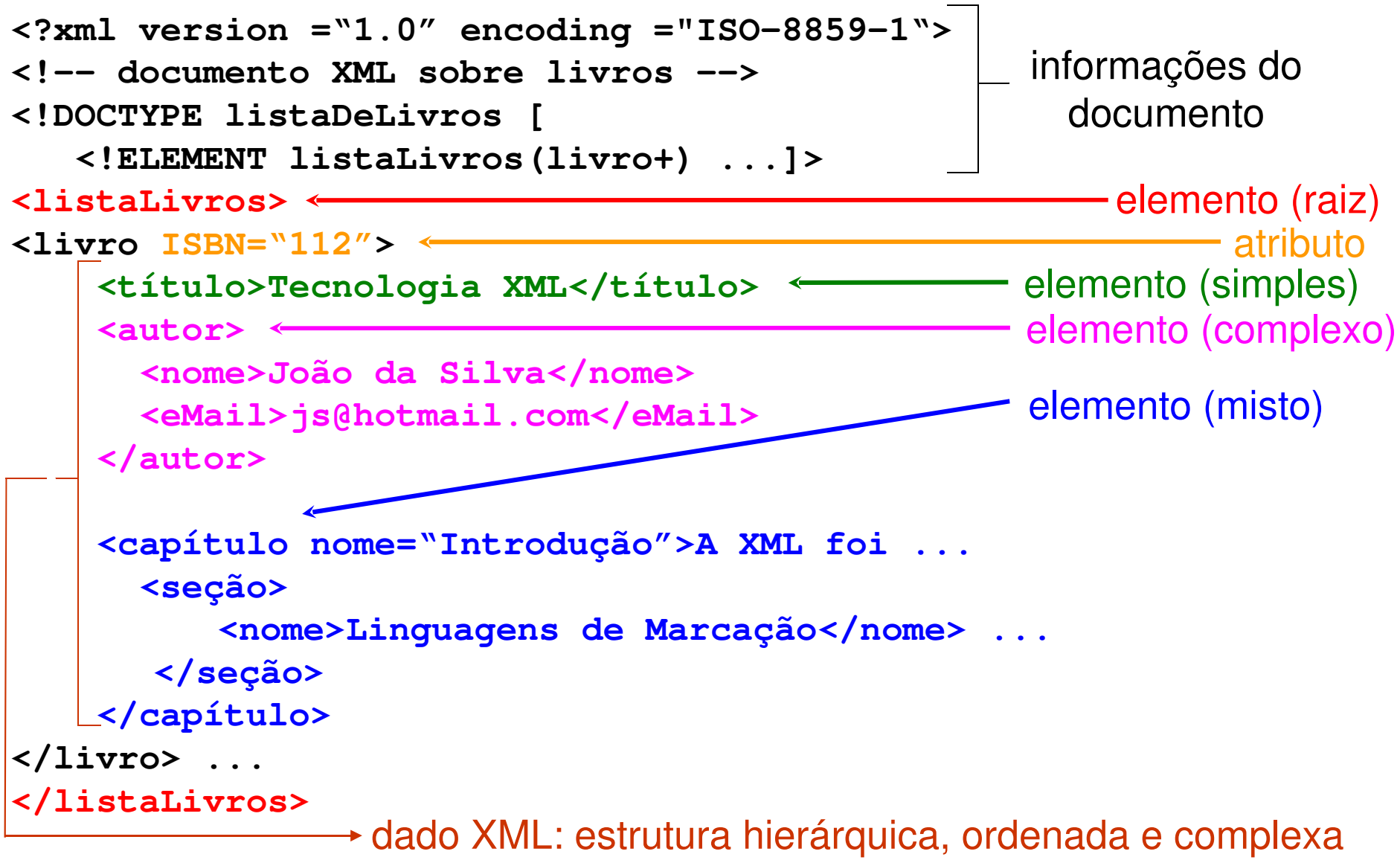
Tecnologia XML x Tecnologia BD

- Conclusão
  - tecnologia XML não é totalmente equivalente à tecnologia de BD
- Tópico de pesquisa na comunidade científica de BD
  - gerenciamento de dados XML através de BDs
  - uma linha de pesquisa
    - desenvolvimento de SGBDs específicos para XML

XML – definição

- XML é uma meta-linguagem de marcação
  - linguagem de marcação
    - semelhante à linguagem HTML
    - utiliza *tags* para descrição os dados
      - ***tag***: indica a intenção do dado e delimita o seu conteúdo
  - meta-linguagem
    - XML é um padrão aberto
      - cada aplicação define o protocolo (linguagem) para a representação dos seus dados
- Dados no formato XML são descritos em um documento XML

Exemplo de Documento XML



Documento XML Bem Formado

- Requisitos
  - contém um elemento raiz
  - define elementos com *tags* inicial e final
  - define atributos com conteúdo delimitado por aspas simples (‘) ou aspas duplas (“)
- *Parser* XML
  - programa que verifica se um documento XML é bem formado
    - alguns *browsers Web* são capazes de realizar tal verificação

Tecnologia XML da W3C

- Principais facilidades similares a SGBDs
  - definição de esquemas
    - DTD e XML Schema
  - linguagens de consulta
    - *XPath* e *XQuery*
  - modelo de representação e API de acesso
    - DOM

Definição de Esquemas

- Esquema XML
  - define restrições para a organização hierárquica e conteúdo dos dados em um doc XML
  - documento válido
    - documento cuja estrutura está de acordo com um esquema
    - validação é feita por um *parser*
- Duas recomendações
  - DTD (*Document Type Definition*)
  - XSD (*XML Schema Definition*)

DTD

- Primeira recomendação da W3C
- Gramática para definição de hierarquia
  - baseada em seqüências ordenadas e escolhas
- Definição de elementos
  - complexos, textuais (`#PCDATA`), vazios (`EMPTY`), mistos (`(#PCDATA | ...)*`) ou com conteúdo aberto (`ANY`)
- Definição de atributos
  - obrigatórios (`#REQUIRED`) opcionais (`#IMPLIED`), fixos (`#FIXED`), valor *default*, enumeração, referência (`ID`, `IDREF(S)`)

DTD - Exemplo

```
<!ELEMENT listaLivros (livro+)>
<!ELEMENT livro (título, preço, autor+,
                 capítulo+)>
<!ATTLIST livro ISBN ID #REQUIRED
                edicaoAnterior IDREF #IMPLIED>
<!ELEMENT título (#PCDATA)>
<!ELEMENT autor (nome, eMail?)>
<!ELEMENT nome (#PCDATA)>
<!ELEMENT preço (#PCDATA)>
<!ELEMENT eMail (#PCDATA)>
<!ELEMENT capítulo (#PCDATA | seção)*>
<!ATTLIST capítulo nome CDATA #REQUIRED>
<!ELEMENT seção (nome, conteúdo)>
<!ELEMENT conteúdo (#PCDATA)>
```

XML Schema (XSD)

- Recomendação mais recente
- Sintaxe XML
- Extensão da funcionalidade de uma DTD
  - definição e especialização de tipos de elementos
    - semelhança com esquemas orientados a objetos
  - definição de tipos de dados
    - simples (*string*, *integer*, *boolean*, ...)
    - complexos (*list*, *union*)
  - facilidades adicionais para definição de restrições
    - intervalos de valores permitidos, padrões de conteúdo via expressões regulares, ...
  - ...

XSD - Exemplo

```
<?xml version="1.0" encoding="UTF-8">
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  ... <!-- Declaração de Tipos -->
  <xsd:simpleType name="Tisbn">
      <xsd:restriction base="xsd:string">
        <xsd:pattern value="[0-9]{2}-[0-9]{3}-[0-9]{4}-[0-9]"/>
      </xsd:restriction>
  </xsd:simpeType>
  <xsd:complexType name="Tlivro">
    <xsd:sequence>
          <xsd:element name="titulo" type="xsd:string"/>
          <xsd:element name="autor" type="Tautor"
            minOccurs="1" maxOccurs="unbounded"/>
          <xsd:element name="preço" type="xsd:float"/>
          ...
    </xsd:sequence>
     <xsd:attribute name="isbn" type="Tisbn"/>
  </xsd:complexType>
  ...
```

XSD – Exemplo (cont.)

```
...
  <xsd:complexType name="TlivroTécnico" base="Tlivro"
                derivedBy="extension">
    <xsd:element name="area" type="xsd:string"
       minOccurs="1" maxOccurs="1"/>
  </complexType>
  ...
  <!-- Declaração de Elementos -->
  <xsd:element name="listaLivros">
     <xsd:complexType>
       <xsd:sequence>
            <xsd:element name="livro" type="Tlivro"
                 minOccurs="1" maxOccurs="unbounded"/>
      </xsd:sequence>
     </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

XPath

- Primeira recomendação para consulta a dados
- Linguagem para o acesso a partes de um doc XML
  - sintaxe: expressões de caminho
    - assemelha-se à navegação em diretórios de arquivos
  - exemplo
    - expressão *XPath*: `/listaLivros/livro/título`
    - resultado:

```
<título>Tecnologia XML</título>
<título>Sistema de Banco de Dados</título>
...
```

XPath - Exemplos

`/listaLivros` (elemento raiz – todo o doc XML)

`/listaLivros/livro/*/eMail` (‘*’ substitui 1 elem)

`/listaLivros/livro//seção` (qq elemento descendente seção)

`/listaLivros/livro/capítulo[1]` (primeiro capítulo de livros)

`/listaLivros/livro/capítulo/nome |`

`/listaLivros/livro/capítulo/seção/nome` (união)

`/listaLivros/livro/@ISBN` (acesso a um atributo)

`/listaLivros/livro[título = “XML”]` (filtro)

`/listaLivros/livro[capitulo/@nome = “XML”`
`or //seção/nome = “XML”]/título` (filtro)

`/listaLivros/livro//seção[last()]` (função)

XQuery

- Recomendação mais recente
- Recursos adicionais em relação à *XPath*
  - junções, definição de estruturas de resultado, variáveis de consulta, atributos calculados, funções de agregação, ...
- Sintaxe básica (expressão "FLWR")

```
for variável in expressãoXPath
[let associação de novas variáveis]
[where condição]
return estrutura de resultado
```

XQuery - Exemplos

```
for $liv in /listaLivros/livro
where $liv/autor/nome = "João Silva"
return { $liv/@ISBN, $liv/titulo }
```

(consulta simples)

```
for $liv in /listaLivros/livro
let $pDesc := $liv/preço - $liv/preço * 0.1
where $liv/categoria = "ficcao"
return <FiccaoDesc>{$liv/titulo, $pDesc}</FiccaoDesc>
```

```
for $liv1 in /listaLivros/livro[@ISBN = "562"]
for $liv2 in /listaLivros/livro
where $liv2/@ISBN != $liv1/@ISBN
and $liv2/autor/nome = $liv1/autor/nome
return $liv2/titulo
```

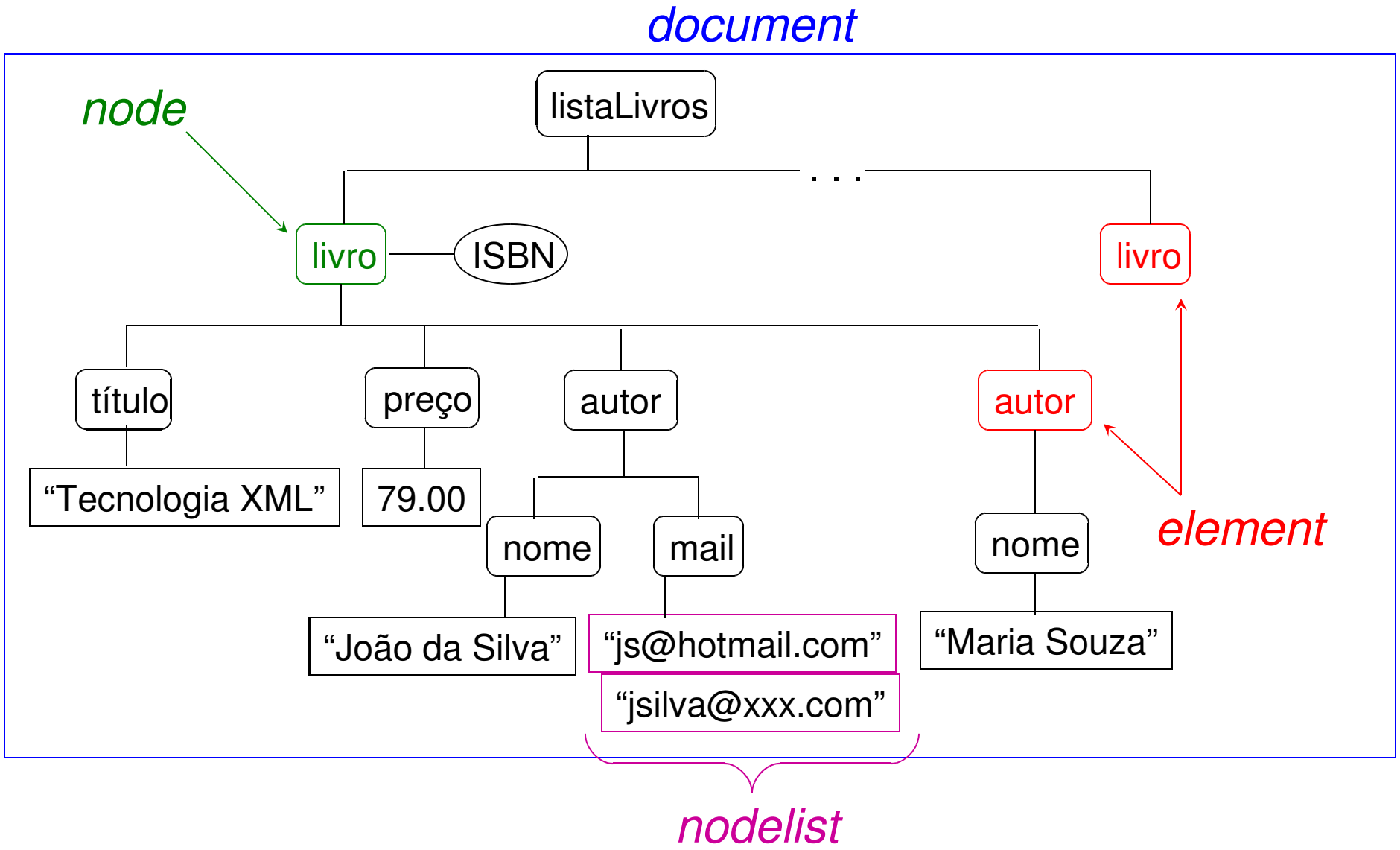
(nova estrutura de resultado)

(junção)

DOM (*Document Object Model*)

- Modelo de dados para XML
  - estrutura hierárquica (árvore)
  - métodos de acesso (API DOM)
    - principais classes de objetos
      - *document*, *node*, *nodelist* e *element*
    - execução de consultas e atualizações de dados
- *Parsers* DOM
  - validam um doc XML
  - geram um objeto *document*

Objetos do Modelo DOM



Principais Métodos

document

Método	Resultado
documentElement	Element
getElementByTagName(String)	NodeList
createTextNode(String)	String
createComment(String)	Comment
createElement(String)	Element

Principais Métodos

node

Método	Resultado
nodeName	String
nodeValue	String
nodeType	short
parentNode	Node
childNodes	NodeList
firstChild	Node
lastChild	Node
previousSibling	Node
nextSibling	Node
insertBefore(Node novo, Node ref)	Node
replaceChild(Node novo, Node antigo)	Node
removeChild(Node)	Node
hasChildNode	boolean

Principais Métodos

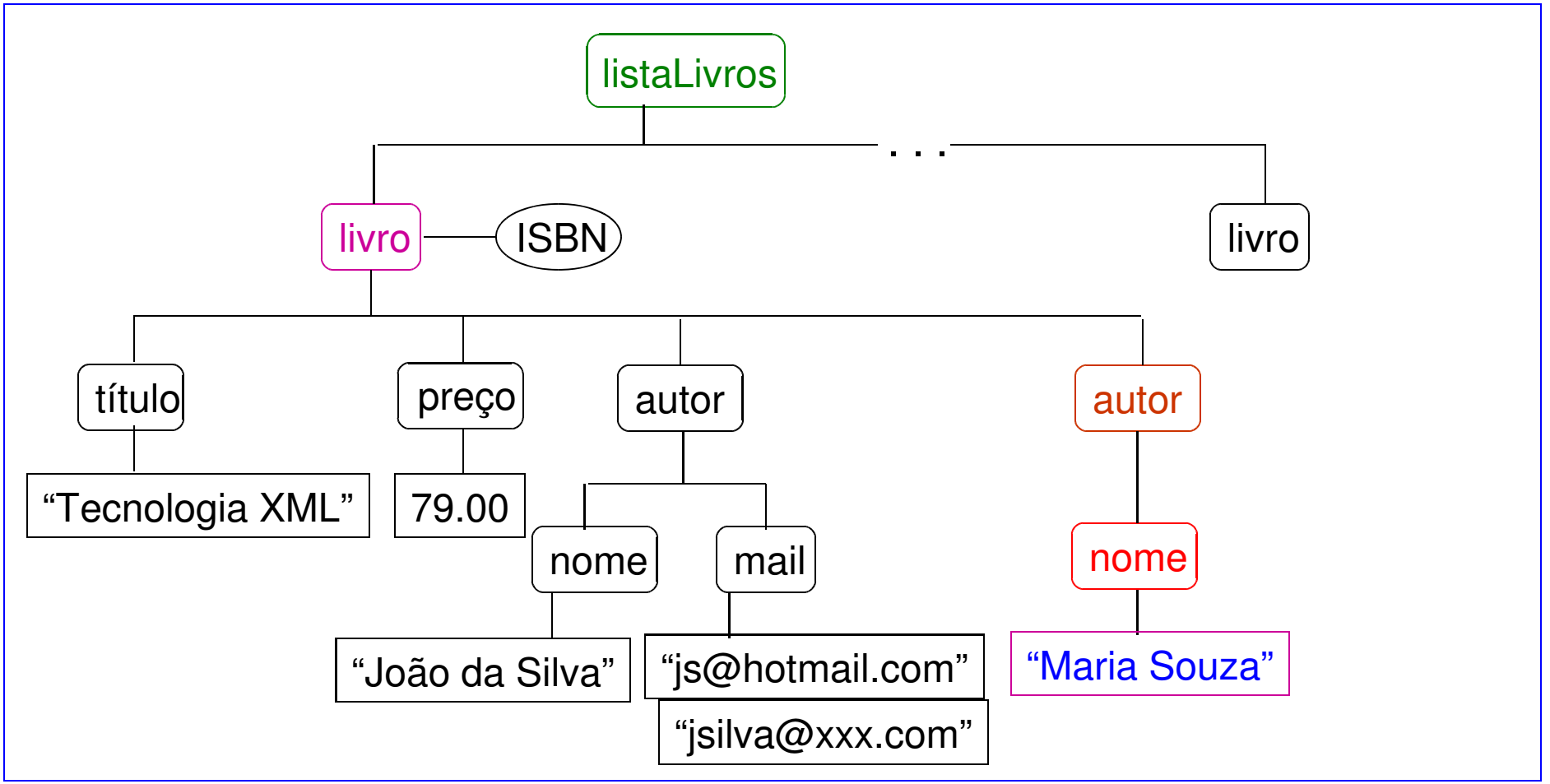
element

Método	Resultado
tagName	String
getAttribute(String)	String
setAttribute(String nome, String valor)	Attr
getAttributeNode(String)	Attr
removeAttributeNode(String)	Attr
getElementsByTagName	NodeList

nodeList

Método	Resultado
Length	int
item(int)	Node

Exemplo de Navegação em DOM



doc.documentElement.childNodes.item(0).getElementsByTagName("autor").item(1).firstChild.firstChild.data

- objeto DOM
- nodo raiz
- lista de livros
- 1º livro
- lista de autores
- 2º autor
- 1º nodo filho: nome
- 1º nodo filho: conteúdo de nome
- texto

DOM – Exemplo (*JavaScript*)

```
var doc, raiz, livro1, autores, autor2;
doc = new ActiveXObject("Microsoft.XMLDOM");
doc.load("livros.xml");
if (doc.parseError != 0) ...;
else
{
   raiz = doc.documentElement;
   /* busca o primeiro livro (primeiro nodo filho) */
   livro1 = raiz.childNodes.item(0);
   /* busca a lista de autores do primeiro livro */
   autores = livro1.getElementsbyTagName("autor");
   /* busca o segundo autor */
   autor2 = autores.item(1);
   /* escreve o nome do autor - primeiro nodo filho */
   document.write("Nome do segundo autor: " +
     autor.childNodes.item(0).data);
}
```