XML
introdução

Helder da Rocha
(helder@argonavis.com.br)

argonavis

Atualizado em setembro de 2001

O que é XML?

- e**X**tensible **M**arkup **L**anguage: padrão W3C
- Uma maneira de **representar** informação
  - não é uma linguagem específica
  - não define vocabulário de comando
  - não define gramática, apenas regras mínimas de estrutura
- Exemplo: documento XML

usuario_33.xml

```
<contato codigo="33">
  <nome>Severino Severovitch</nome>
  <email>bill@norte.com.br</email>
  <telefone tipo="celular">
    <area>11</area>
    <numero>9999 4321</numero>
  </telefone>
</contato>
```

elemento

atributo

"nó" de texto

XML versus HTML

HTML mostra ***como*** *apresentar*

```
<h1>Severino Severovitch</h1>
<h2>bill@norte.com.br</h2>
<p>
  <b>11</b>
  <i>9999 4321</i>
</p>
```

XML mostra ***o que*** *significa*

```
<nome>Severino Severovitch</nome>
<email>bill@norte.com.br</email>
<telefone>
  <ddd>11</ddd>
  <numero>9999 4321</numero>
</telefone>
```

Anatomia de um documento XML

- *Documentos XML são documentos de texto Unicode*
  - *É uma hierarquia de **elementos** a partir de uma **raiz***
  - *Menor documento tem um elemento (vazio ou não):*

```
<nome> Северино Северович </nome>
```

Elemento raiz

  - *Menor documento contendo elemento vazio*

```
<nome></nome>   =   <nome/>
```

  - *Menor documento contendo elemento e conteúdo texto*

```
<nome>  Северино Северович  </nome>
```

Etiqueta inicial — *Conteúdo do Elemento* — *Etiqueta final*

XML Namespaces

- *Estabelecem um contexto para elementos e atributos*
  - *É formalmente declarado através de um identificador (um string, geralmente uma URI) através de atributo reservado do XML:* ***xmlns***
- *Podem ser associados a um* ***prefixo*** *para qualificar elementos e atributos*
  - *Quando o prefixo não é usado, estabelece um* ***namespace default*** *adotado pelo elemento onde é declarado e seus elementos filho*

```
<simulacao>
  <tempo unidade="segundos">130</tempo>
  <clima xmlns="uri://app-clima">
    <tempo>chuvoso</tempo>
  </clima>
</simulacao>
```

Escopo do namespace vale para elemento ***<clima>*** *e* ***herdado*** *por todos os seus descendentes*

Escopo do namespace vale para descendentes de ***<simulacao>*** *qualificados com o* ***prefixo 'w'***

```
<simulacao xmlns:w="uri://app-clima">
  <tempo unidade="segundos">130</tempo>
  <w:clima>
    <w:tempo>chuvoso</w:tempo>
    <tempo unidade="horas">2.5</tempo>
  </w:clima>
</simulacao>
```

Nos dois casos, elementos ***<tempo>*** *significam coisas diferentes, mas não há conflito porque pertencem a namespaces diferentes (um deles não tem namespace)*

Documentos XML bem formados

- Para que possa ser manipulado como uma árvore, um documento XML precisa ser **bem formado**
  - Documentos que não são bem formados não são documentos XML – use um editor XML para descobrir
- Documentos bem-formados obedecem as regras de construção de documentos XML genéricos
- Regras incluem
  - Ter um, e apenas um, elemento raiz
  - Valores dos atributos estarem entre aspas ou apóstrofes
  - Atributos não se repetirem
  - Todos os elementos terem etiqueta de fechamento
  - Elementos estarem corretamente aninhados

Exemplos de algumas regras

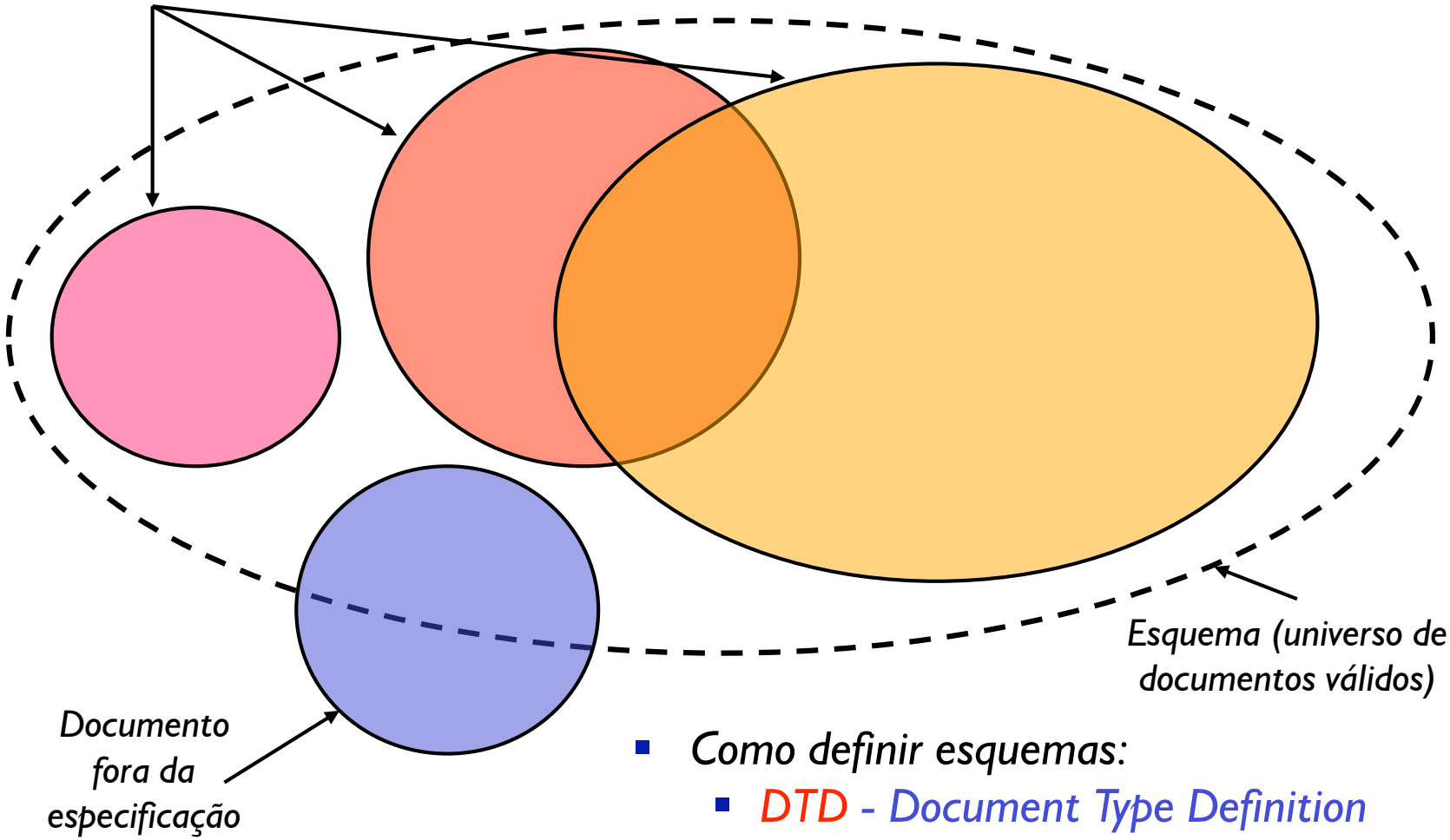
- *(1) Elementos não devem se sobrepor*
  - *Não pode* `<a><b></a></b>`
- *(2) Atributos têm que ter valores entre aspas*
  - *Não pode* `<hr width=10>`
  - *Deve ser* `<hr width="10">`
- *(3) Nomes de elementos são case-sensitive*
  - *Não pode* `<a>...</A>`
- *(4) Todos os elementos têm marcadores de abertura e de fechamento*
  - *Não pode* `<br>x<br>`
  - *Deve ser* `<br/>x<br/>` *ou* `<br></br>x<br></br>`

XML válido

- Um XML bem construído pode não ser **válido** em determinada aplicação
- Aplicação típica pode esperar que
  - elementos façam parte de um **vocabulário** limitado,
  - certos atributos tenham **valores** e **tipos** definidos,
  - elementos sejam organizados de acordo com uma determinada **estrutura hierárquica**, etc.
- É preciso **especificar** a linguagem!
  - **Esquema**: modelo que descreve todos os elementos, atributos, entidades, suas relações e tipos de dados
- Um documento é considerado válido **em relação a um esquema** se obedecer todas as suas regras

Esquema

- *O esquema representa uma **classe***
- *Os documentos são **instâncias***



- *Como definir esquemas:*
  - *DTD - Document Type Definition*
  - *W3C XML Schema*

DTD vs. XML Schema

- *Um esquema é essencial para que haja* ***comunicação*** *usando XML*
  - *Pode ser estabelecido "informalmente" (via software)*
  - *Uso formal permite validação usando ferramentas genéricas de manipulação de XML*
- *Soluções padrão do W3C*

DTD

```
<!ELEMENT contato
          (nome, email, telefone)>
<!ATTLIST contato
          codigo NMTOKEN #REQUIRED>
```

- *Simples mas não é XML*
- *Não suporta namespaces*
- *Limitado quando a tipos de dados*

XML Schema

```
<xsd:schema
     xmlns:xsd="../XMLSchema">
 <xsd:element name="contato">
  <xsd:complexType>
   <xsd:attribute name="codigo"
                  use="required">
```

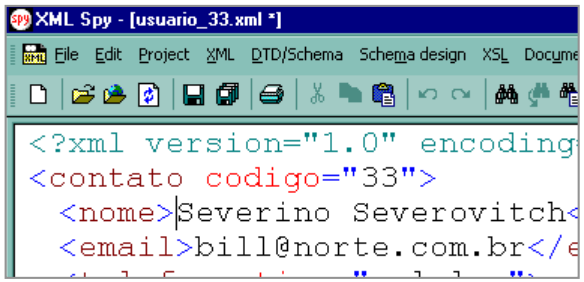
- *É XML, porém mais complexo*
- *Suporta namespaces*
- *Permite definição de tipos*

Por que usar XML para compartilhar dados?

- Porque é um **padrão aberto**
  - Facilidade para converter para formatos proprietários
- Porque é **texto**
  - Fácil de ler, fácil de processar, menos incompatibilidades
- Porque promove a **separação** entre estrutura, conteúdo e apresentação
  - Facilita geração de dados para visualização dinâmica
  - Evita repetição de informação / simplifica manutenção
- Porque permite **semântica** na Web
  - Elementos HTML não carregam significado, apenas dicas de formatação: mecanismos de busca ficam prejudicados

Como produzir XML

- ***Criando*** *um documento de texto Unicode a partir de qualquer editor de textos*



```
<contato codigo="33">
  <nome>Severino Severovitch</nome>
  <email>bill@norte.com.br</email>
  <telefone tipo="celular">
    <area>11</area>
    <numero>9999 4321</numero>
  </telefone>
</contato>
```

- ***Gerando*** *um documento a partir de uma árvore montada dinamicamente*

```
<contato codigo="33">
  <nome>Severino Severovitch</nome>
  <email>bill@norte.com.br</email>
  <telefone tipo="celular">
    <area>11</area>
    <numero>9999 4321</numero>
  </telefone>
</contato>
```

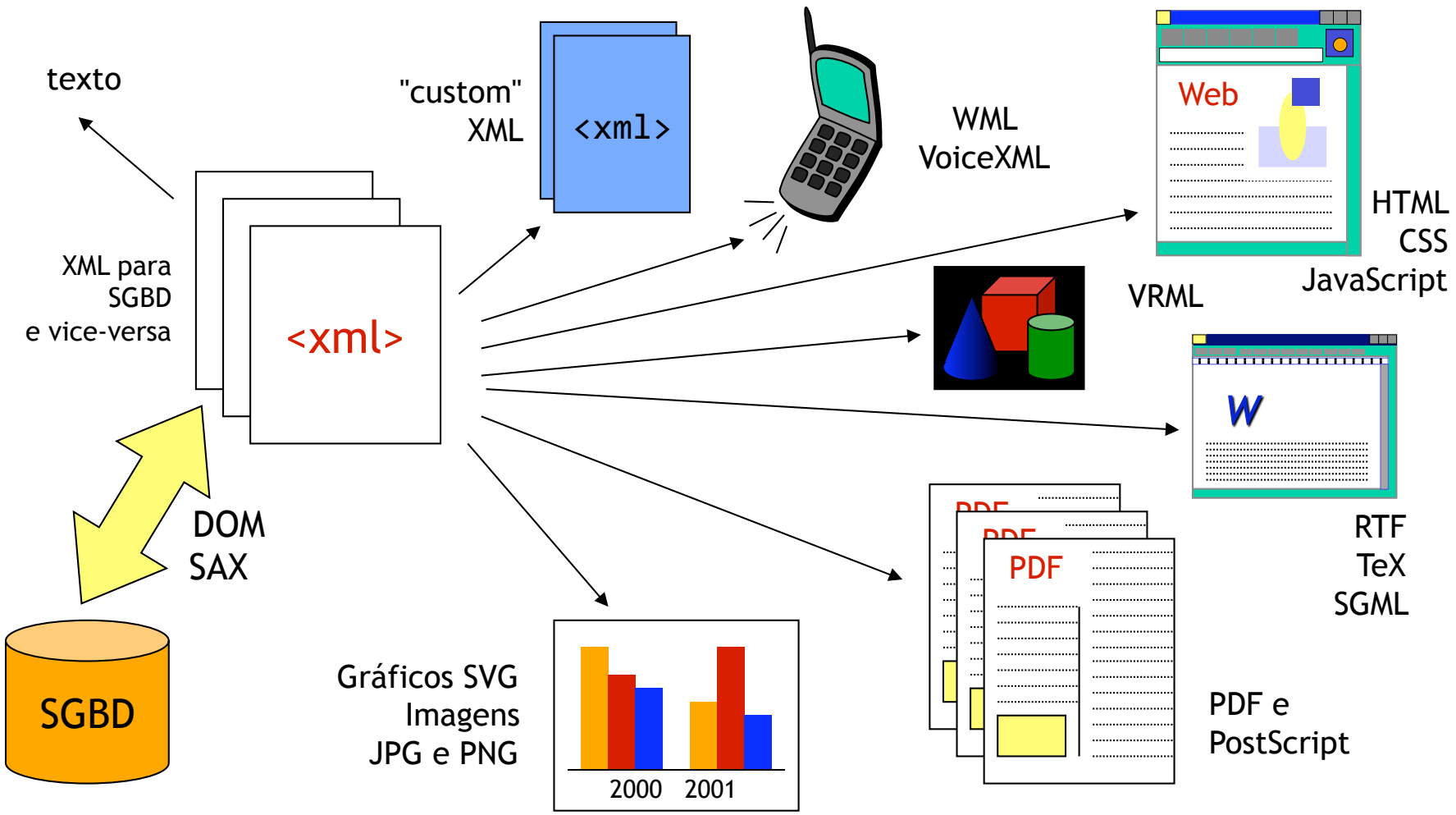
Visualização em um browser

- *Folha de estilo*: conjunto de regras para formatar ou transformar as informações de um documento XML
- **CSS** - Cascading Style Sheets
  - Transformação visando apresentação visual
  - Aplicação do estilo em tempo de execução no cliente
- **XSLT** - eXtensible Stylesheet Language
  - Transformação em texto, HTML ou outro formato
  - Aplicação em tempo real ou prévia (no servidor)
- Se não estiver associado a uma folha de estilo, o documento XML não tem uma "aparência" definida
  - Vários browsers por default mostram a árvore-fonte XML
  - Outros mostram apenas os nós de texto sem formatação

Formas de processamento XML

- Via APIs de programação
  - **SAX** – Simple API for XML: leitura seqüencial, ideal para extração de dados
  - **DOM** – Document Object Model: leitura completa, ideal para manipulação (inserção, reordenação, alteração, remoção de nós)
- Via linguagens de processamento (suportadas por parsers e processadores padronizados pela W3C)
  - **XSLT**, **XPath**, **XLink**, **XPointer** e **XQuery**: extração, transformação e localização de dados
  - **XSL-FO**, **XHTML**, **SVG**: apresentação de dados

Processamento XML



XPath

- *Linguagem usada para navegar na árvore XML*
  - *Uma expressão XPath é um caminho na árvore que resulta em um valor (número, texto, booleano), objeto (elemento, atributo, nó de texto) ou conjunto de objetos*

- *Caminhos absolutos*

```
/contato/@codigo
/contato/telefone/@tipo
/contato/telefone/numero/text()
```

- *Relativos ao contexto* `/contato` :

```
@codigo (ou ./@codigo)
telefone/@tipo (ou ./telefone/@tipo)
telefone/numero/text()
```

- *Expressões XPath são usadas dentro de atributos XML*
  - *Usadas em XSLT, XLink, XQuery e XPointer*

XLink, XPointer e XQuery

- **XLink**: *especificação W3C que define vínculos (de diversos tipos) entre documentos XML*
  - *Funcionalidade mínima é igual ao <a href> do HTML*
  - *É uma coleção de atributos, com namespace próprio, que podem ser usados em elementos de qualquer linguagem XML.*
- **XPointer**: *aponta para partes de documentos XML*
  - *Identificador no destino, acessível por XLink:* `xlink:href="#ident"`
  - *Caminho resultante de expressão XPath:* `xpointer(/livro/id)`
- **XQuery**: *linguagem para pesquisar documentos XML através de queries com sintaxe inspirada em SQL*
  - *Exemplo:*

```
FOR $b IN document("usuario_33.xml")/contato
        WHERE nome="Severino Severovitch"
        RETURN $b
```

XSL

- e**X**tensible **S**tylesheet **L**anguage
  - Aplicação de XML para transformação e apresentação de dados disponíveis em XML
- Não existe mais como **uma** especificação. Foi dividida em duas
  - **XSLT** – foca em transformação de dados. XSLT é uma linguagem funcional para processamento de templates
  - **XSL-FO** – linguagem de descrição de página (como PDF, HTML+CSS)
- Cada especificação define um namespace próprio
  - XSLT usa o identificador `http://www.w3.org/1999/XSL/Transform`
  - XSL-FO usa `http://www.w3.org/1999/XSL/Format`
- Importante: as URLs acima são usadas como **identificadores de namespace**, e não representam nenhum endereço Web
  - É muito comum usar URLs como identificadores de namespace
  - **Todas** as aplicações XML do W3C usam URLs da forma: **`http://www.w3.org/`***`<ano-de-criacao>`*`/`***`<`***`tecnologia`***`>`**

```
<xsl:stylesheet version="1.0"
     xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
   <xsl:template match="livro/titulo">
      <td><xsl:value-of select="." /></td>
```

XSLT

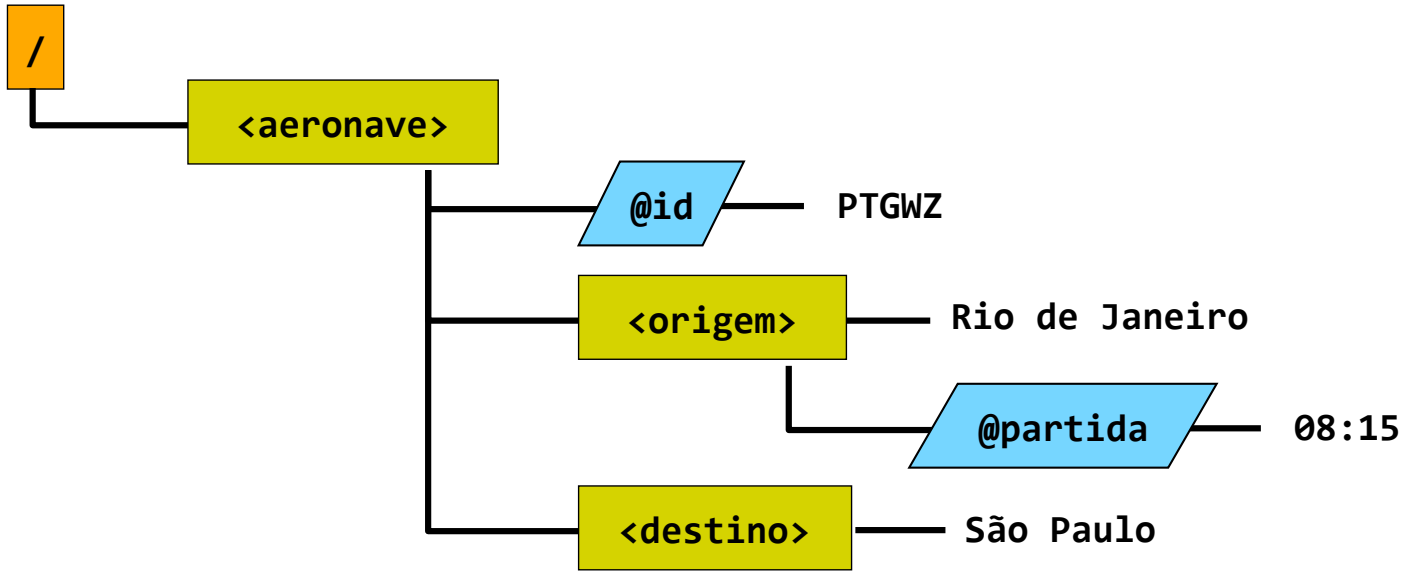
- *XSL Transformations (XSLT)*
  - *Linguagem (XML) para criação de documentos que contêm regras de transformação para documentos XML*
  - *Documentos escritos em XSLT são chamados de **folhas de estilo** (mas são na verdade templates) e contêm*
    - *Elementos XSLT:* `<template>`, `<if>`, `<for-each>`, ...
    - *Expressões XPath para localizar nós da árvore-fonte*
    - *Texto ou XML a ser gerado no documento-resultado*
  - *Usa-se um processador XSLT*

XSLT: documento-fonte (I)

- *Considere o seguinte documento-fonte:*

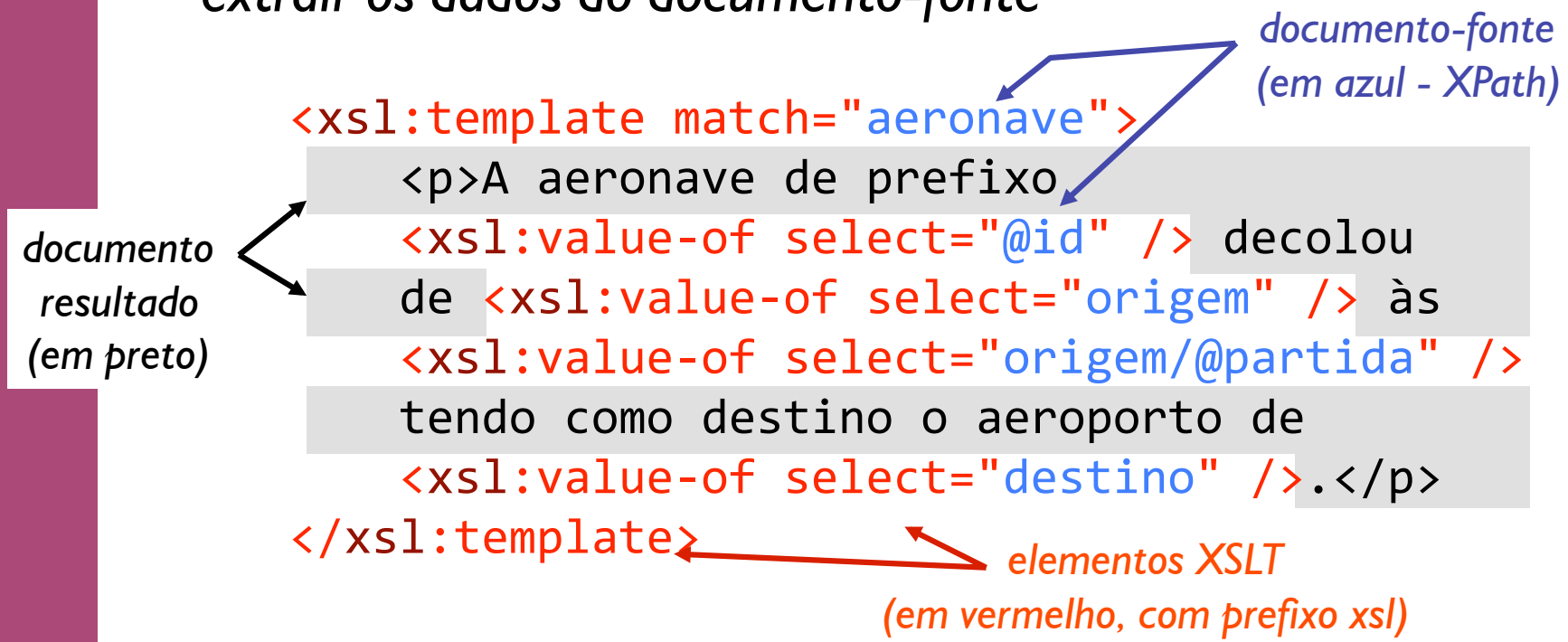
```
<aeronave id="PTGWZ">
   <origem partida="08:15">Rio de
                           Janeiro</origem>
   <destino>São Paulo</destino>
</aeronave>
```

- *E sua representação como uma árvore-fonte*



XSLT: folha de estilos (2)

- *O seguinte **template** (parte de uma folha de estilos XSLT) pode extrair os dados do documento-fonte*



- *Elementos XSLT são qualificados com **prefixo** (da forma `<xsl:elemento>`) para evitar conflitos com o documento-resultado*
  - *O prefixo **xsl** e **namespace** precisam ser declarados com **xmlns:xsl***

XSLT: documento-resultado (3)

- *Após a transformação, o resultado será*

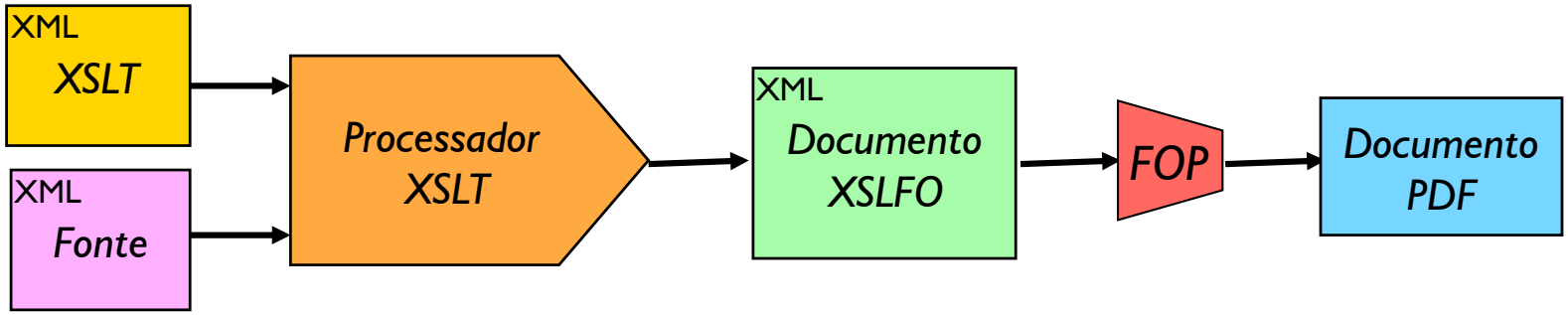
```
<p>A aeronave de prefixo
PTGWZ decolou
de Rio de Janeiro às
8:15
tendo como destino o aeroporto de
São Paulo.</p>
```

- *Para obter outros resultados e gerar outros formatos com os mesmos dados, pode-se criar folhas de estilo adicionais*

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
   <fo:layout-master-set>
      <fo:simple-page-master master-name="p1">
         <fo:region-body/>
```

XSL-FO

- ***XSL Formatting Objects***
  - *Linguagem XML de **descrição de página** com os mesmos recursos que PostScript ou PDF*
  - *Descreve o **layout** preciso de texto e imagens*
  - *Possui centenas de elementos, atributos e propriedades (que são semelhantes às propriedades do CSS)*
  - *Páginas são facilmente convertidas para PDF e PostScript*
  - *Ideal para gerar documentos para impressão (livros, etc.)*
- *Normalmente **gerada** via XSLT*



XSL-FO: menor documento

```
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">

   <fo:layout-master-set>
      <fo:simple-page-master master-name="p1">
         <fo:region-body/>
      </fo:simple-page-master>
   </fo:layout-master-set>

   <fo:page-sequence master-name="p1">
      <fo:flow flow-name="xsl-region-body">
         <fo:block color="blue" font-size="20pt">
            Hello PDF!
         </fo:block>
      </fo:flow>
   </fo:page-sequence>
</fo:root>
```

Este é o "<head>" do XSL-FO

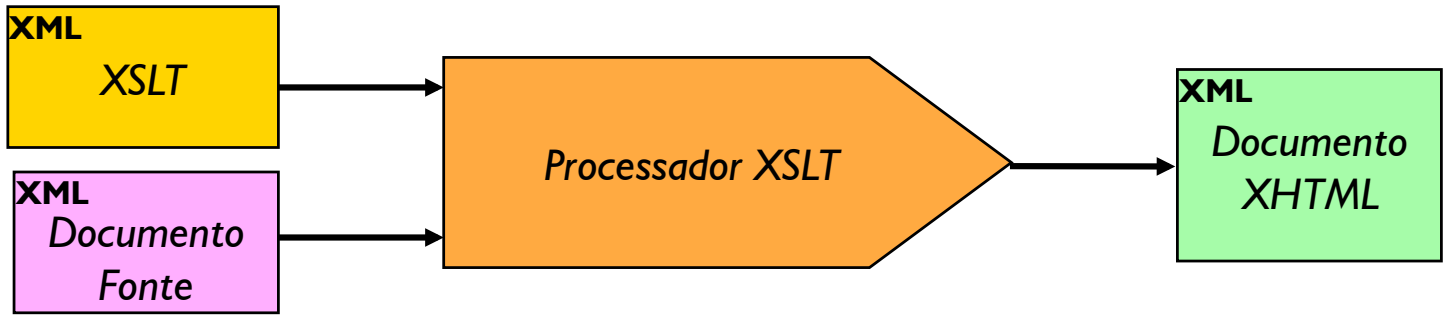
Ligação entre as regras de layout e o conteúdo afetado

Este é o "<body>" do XSL-FO

```
<html xmlns="http://www.w3.org/1999/xhtml">
   <head><title>Página XHTML</title></head>
   <body>
      <h1>Página XHTML</h1>
```

XHTML

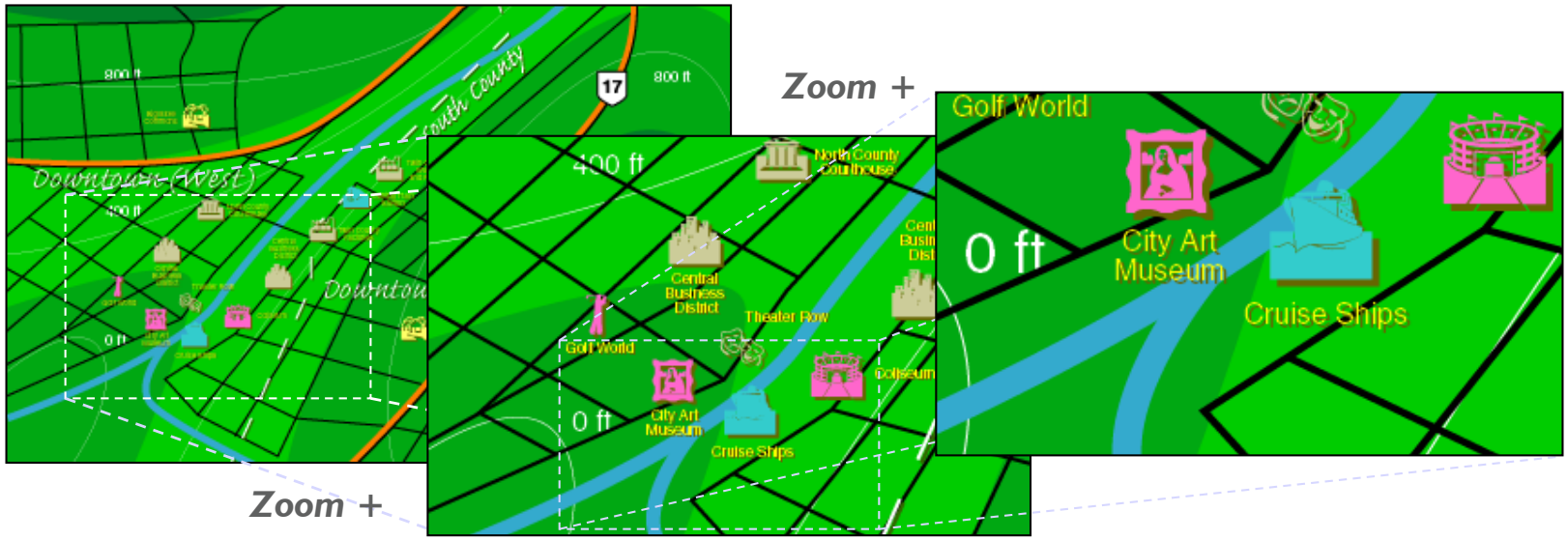
- e**X**tensible **HTML**
  - *Linguagem XML de* ***descrição de página Web***
  - *Mesmos elementos do HTML 4.0 Strict*
  - *Elementos descrevem somente a estrutura dos componentes da página. A forma precisa ser especificada usando CSS: não há elementos/atributos para mudar cor, alinhamento, etc.*
  - *Pode ser misturada (estendida) com outras linguagens XML (MathML, SVG, linguagens proprietárias)*
- *Normalmente* ***gerada*** *via XSLT*



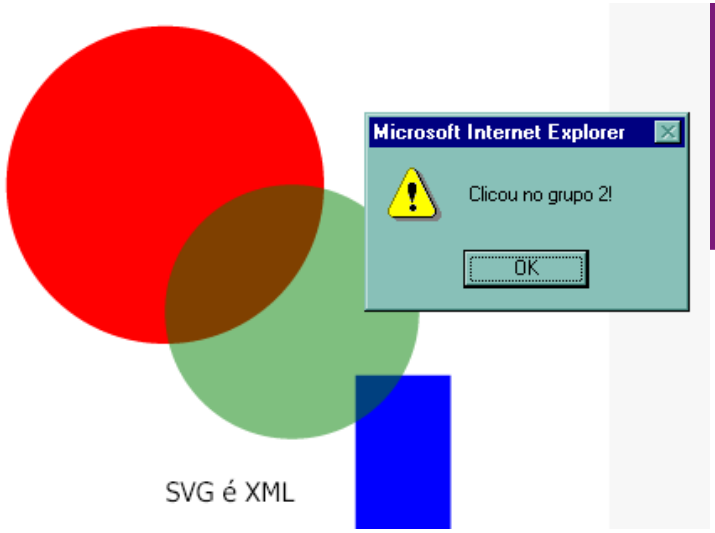
```
<svg xmlns="http://www.w3.org/2000/svg">
   <circle style="fill: red" cx="3cm" cy="3cm" r="2.5cm" />
   <rect style="fill: blue" x="6cm" y="6cm"
         height="2.5cm" width="1.5cm" />
```

SVG

- *Scalable Vector Graphics (padrão W3C)*
  - *Gráficos vetoriais em XML*
  - *Plug-ins para principais browsers: concorre com Flash*
  - *Suporta animações, links, JavaScript, CSS*
  - *Produzido por ferramentas como Adobe Ilustrator*
  - *Pode ser embutido no código XHTML e XSL-FO*



Exemplo de SVG



JavaScript

CSS

XLink

```
<svg width="10cm" height="10cm">
  <g onclick="alert('Clicou no grupo 1!')">
    <circle style="fill: red"
            cx="3cm" cy="3cm" r="2.5cm" />
    <rect style="fill: blue" x="6cm" y="6cm"
          height="2.5cm" width="1.5cm" /></g>
  <g onclick="alert('Clicou no grupo 2!')">
    <circle style="fill: green; opacity: 0.5"
            cx="5cm" cy="5cm" r="2cm" /></g>
  <a xmlns:xlink="http://www.w3.org/1999/xlink"
     xlink:href="http://www.w3.org/Graphics/SVG">
    <text style="color: black; font-family: tahoma;
                 font-size: 12pt" x="3cm" y="8cm">
    SVG é XML</text></a>
</svg>
```

Outras aplicações populares do XML

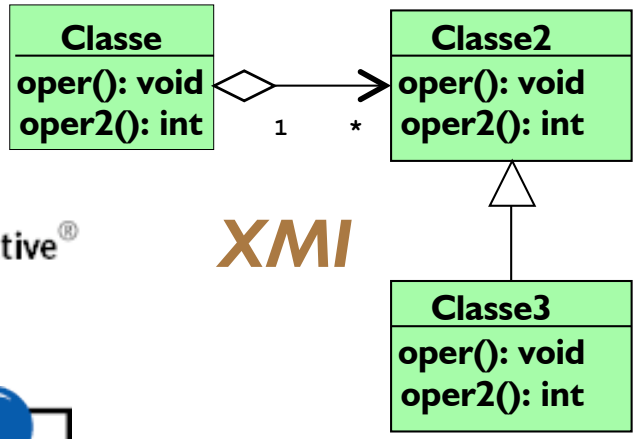
$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

MathML

TEI

TEXT ENCODING INITIATIVE

Dublin Core Metadata Initiative®



XMI

WML

VoiceXML

Web Services

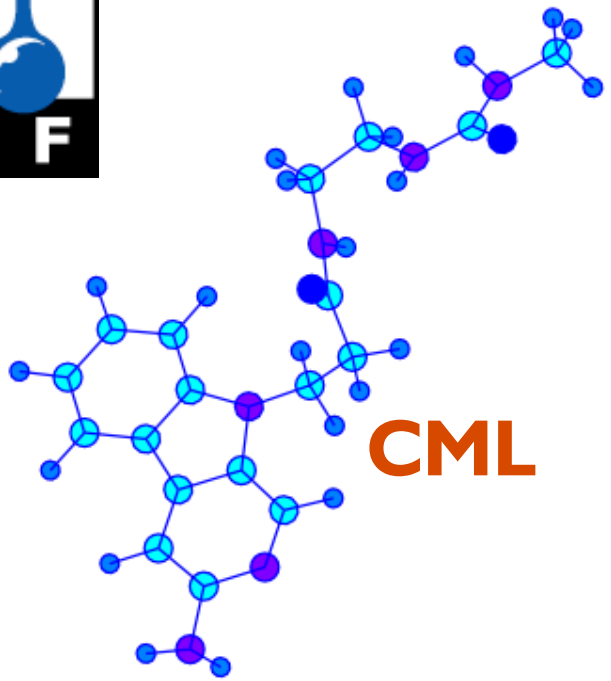
SOAP

WSDL

XML-RPC

UDDI

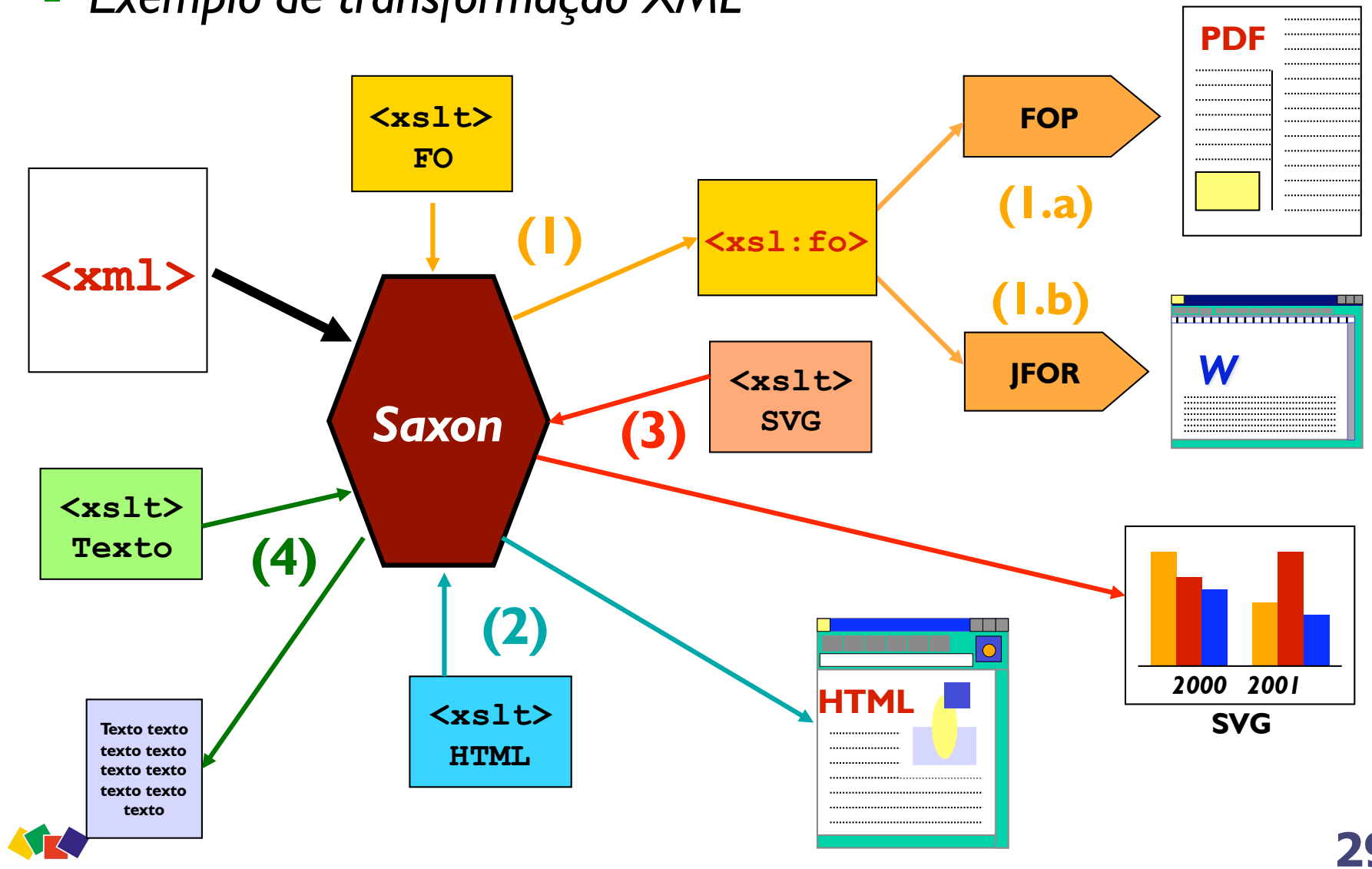
ebXML



CML

Demonstração

- *Exemplo de transformação XML*



Conclusões

- *XML é uma ótima solução para **compartilhar** dados*
- *Para **implementar** soluções em gestão de informações usando XML, pode-se usar*
  - *DTD ou XSchema para especificar o modelo de dados e validar as informações*
  - *As APIs DOM ou SAX para extrair dados dos documentos, gerar documentos, ler e gravar em bancos de dados*
  - *XSLT e XPath para transformar os dados em outros formatos*
  - *XLink, XPointer e XQuery para criar vínculos lógicos entre os documentos e localizar seus componentes*
  - *XSL-FO ou XHTML para formatar os dados para impressão ou visualização na tela (PDF, Word ou Web)*
  - *SVG para gerar informações em forma de gráfico vetorial*