XML

na gestão de informações

Helder da Rocha
Consultor

Objetivos

- *Responder às questões*
  - ***Como implementar*** *soluções de gestão de informações usando XML?*
  - ***Quando e como usar*** *as tecnologias e linguagens que viabilizam o compartilhamento de informações?*
- *Apresentar*
  - *Breve* ***introdução ao XML*** *e tecnologias relacionadas.*
  - *Recursos para manipular informações representadas em XML:* ***ferramentas, linguagens e tecnologias***

Programa

Por que XML? Onde usar XML?
Como produzir documentos XML
Documentos válidos: DTD e XML Schema

Manipulação via programação em DOM e SAX
Transformação: XSLT e XPath
Localização e extração: XLink, XQuery e XPointer
Visualização: XSL-FO e SVG

Demonstração: geração de HTML, RTF e PDF

Ferramentas e Conclusões

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 comandos
  - não define uma gramática, apenas regras mínimas
- 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*

Partes de um documento

elemento raiz

declaração XML

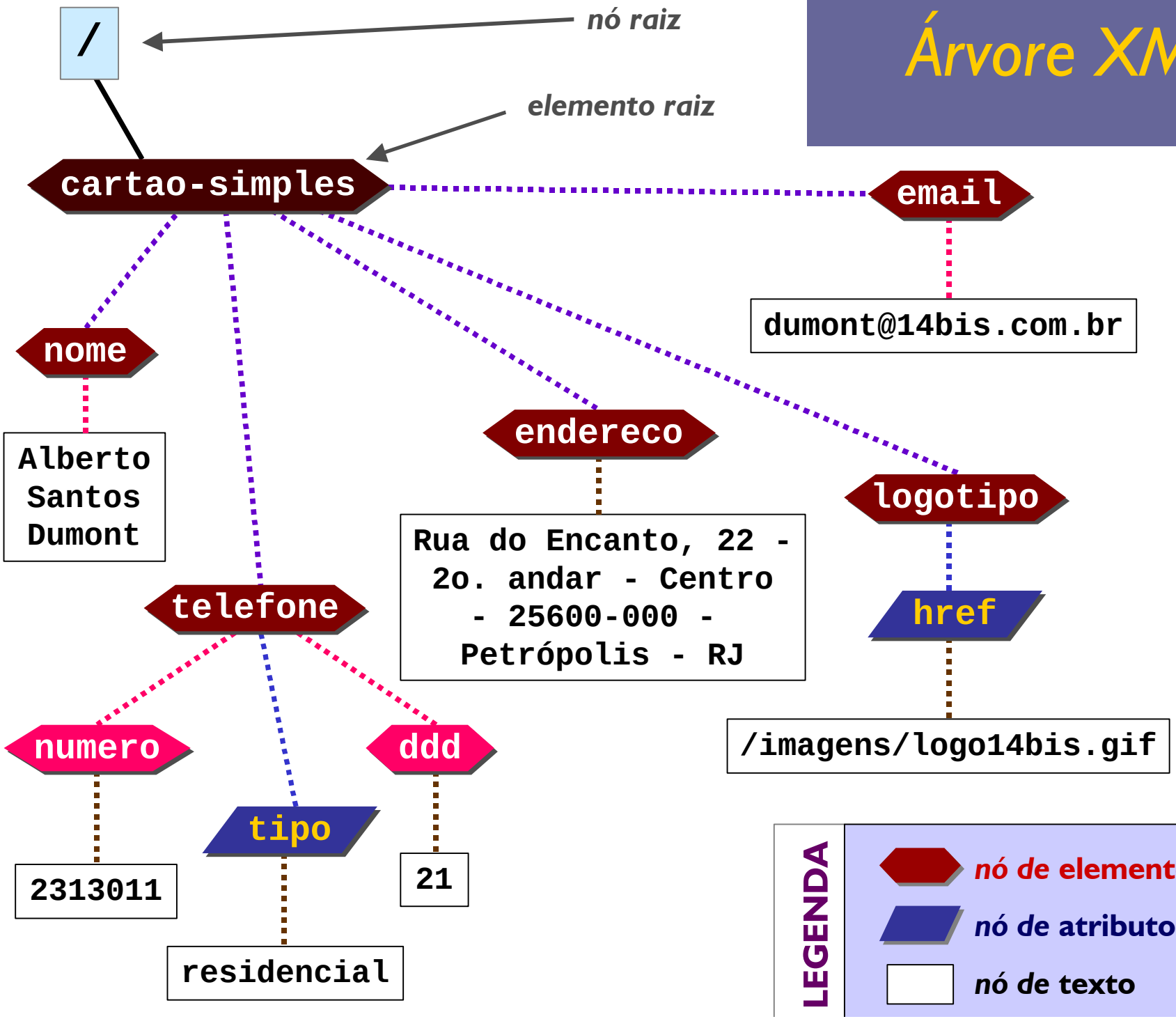
nó raiz (/)

```
<?xml version="1.0" encoding="iso-8859-1" ?>
<cartao-simples>
   <logotipo href="/imagens/logo14bis.gif" />
   <nome>Alberto Santos Dumont</nome>
   <endereco>Rua do Encanto, 22 - 2o. andar -
   Centro - 25600-000 - Petrópolis - RJ</endereco>
   <email>dumont@14bis.com.br</email>
   <telefone tipo="residencial">
      <ddd>21</ddd>
      <numero>2313011</numero>
   </telefone>
</cartao-simples>
```

atributos

elementos

Árvore XML


/
nó raiz
elemento raiz
cartao-simples
email
dumont@14bis.com.br
nome
Alberto Santos Dumont
endereco
Rua do Encanto, 22 - 2o. andar - Centro - 25600-000 - Petrópolis - RJ
logotipo
href
/imagens/logo14bis.gif
telefone
numero
2313011
tipo
residencial
ddd
21
LEGENDA
nó de elemento
nó de atributo
nó de texto

XML Namespaces

- *Limita o escopo de elementos*
  - *Evita conflitos quando duas linguagens se cruzam no mesmo documento*
- *Consiste da associação de um **identificador** a cada elemento/atributo da linguagem, que pode ser*
  - *herdado através do escopo de uma sub-árvore*
  - *atribuído explicitamente através de um prefixo*
- *Exemplo*

prefixo *identificador*

```
<cadastro xmlns:firma="01.234.567/0001-99">
  <nome>Severino Severovitch</nome>
  <firma:nome>Sibéria Informática Ltda.</firma:nome>
  <email>bill@norte.com.br</email>
</cadastro>
```

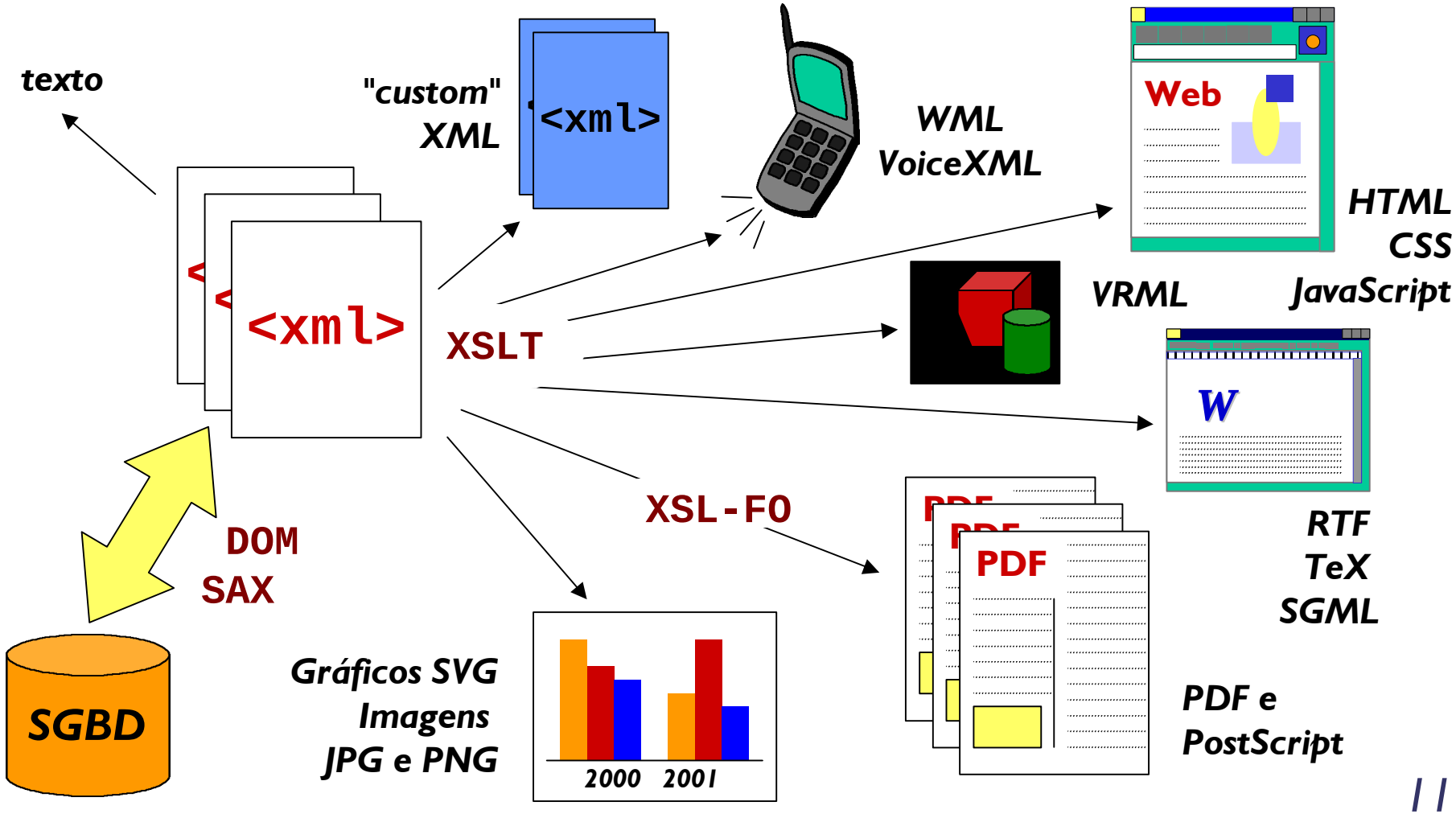
Este elemento <nome> pertence a outro namespace

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 permitirá **semântica** na Web
  - Elementos HTML não carregam significado, apenas dicas de formatação: mecanismos de busca ficam prejudicados
  - Solução com XML dependerá de suporte dos clientes

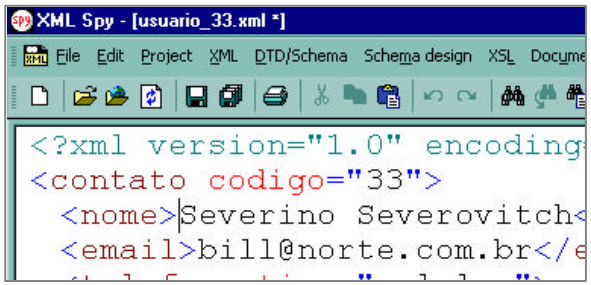
Onde usar XML?

- *Dados armazenados em XML podem ser facilmente transformados em outros formatos*



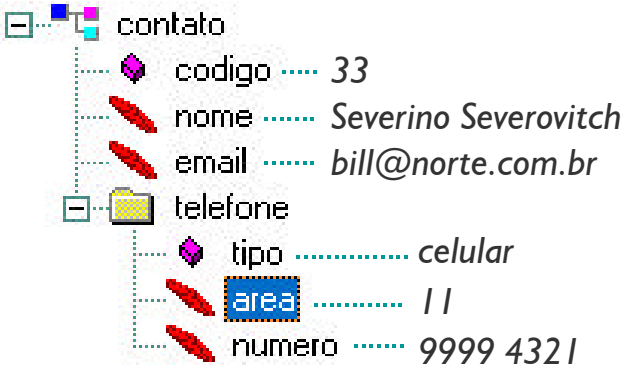
Como produzir XML

- **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>
```

Documentos XML bem formados

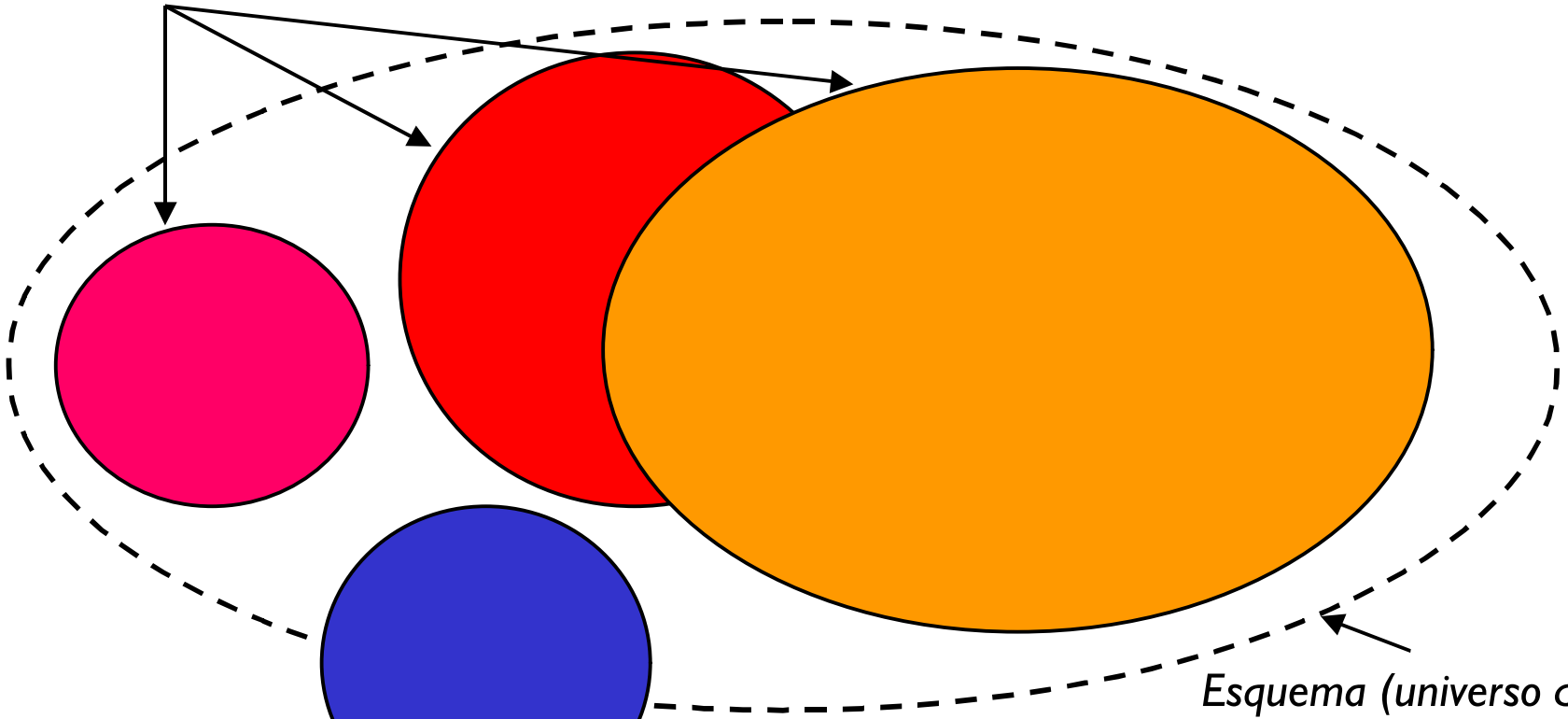
- Para que possa ser manipulado como uma árvore, um documento XML precisa ser **bem formado**
  - Documentos que não são bem formados não são documentos XML
- Documentos bem-formados obedecem as regras de construção de documentos XML genéricos
- Regras incluem
  - Ter um, e apenas um, elemento raiz
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

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 XML 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*

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

XSchema

```
<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

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
  - Internet Explorer e outros mostram a árvore-fonte XML
  - Netscape mostra apenas os nós de texto

Como manipular XML?

- Há duas APIs padrão para manipular (interpretar, gerar, extrair dados e tratar eventos) arquivos XML:
  - W3C Document Object Model (W3C DOM)
  - Simple API for XML (SAX)
- Servem a finalidades diferentes
- Implementações disponíveis em várias linguagens
- **SAX** oferece métodos que respondem a **eventos** produzidos durante a leitura do documento
  - notifica quando um elemento abre, quando fecha, etc.
- **DOM** monta uma **árvore**, que permite a navegação na estrutura do documento
  - propriedades dos objetos podem ser manipuladas

Leitura de XML com SAX

- *Se um processador SAX receber o documento ...*

```
<carta>
  <mensagem id="1">Bom dia!</mensagem>
</carta>
```

- *... ele irá disparar a seguinte seqüência de eventos:*

```
startDocument()
  startElement("carta", [])
    startElement("mensagem", [Attribute("id","1")])
      characters("Bom dia!")
    endElement("mensagem")
  endElement("carta")
endDocument()
```

- *Programador deve implementar um objeto "ouvinte" para capturar os eventos e extrair as informações desejadas*

Criação de documentos com DOM (I)

- *Criação dos elementos*

/	Document	*Obter objeto do tipo* ***Document*** *(raiz) (dependente de processador):* **doc**
<carta>	Element	`carta = doc.createElement("carta")`
<mensagem>	Element	`mens = doc.createElement("mensagem")`
Bom dia!	String	`texto = doc.createTextNode("Bom dia!")`

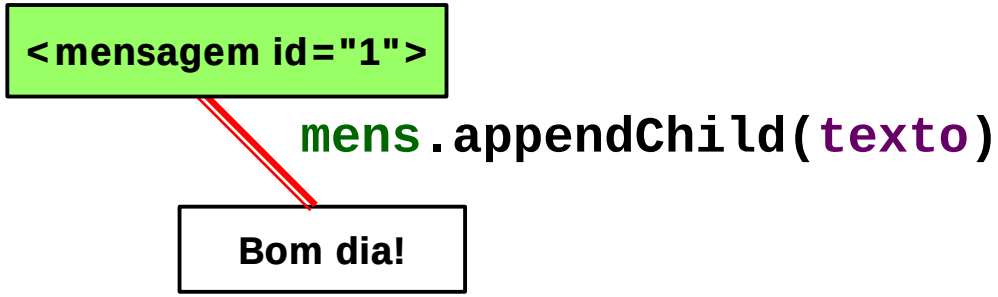
- *Atributos*

<mensagem id="1">	`mens.setAttribute("id", "1")`

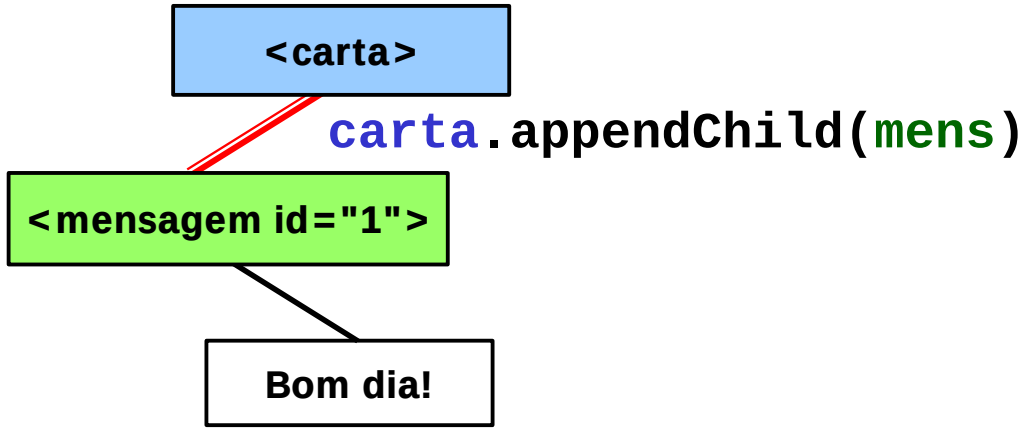
Criação de documentos com DOM (2)

Montagem da árvore passo-a-passo

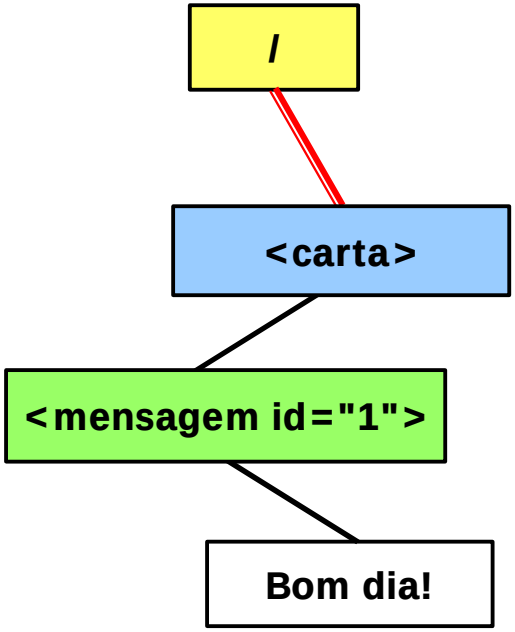
1. Sub-árvore <mensagem>



2. Sub-árvore <carta>



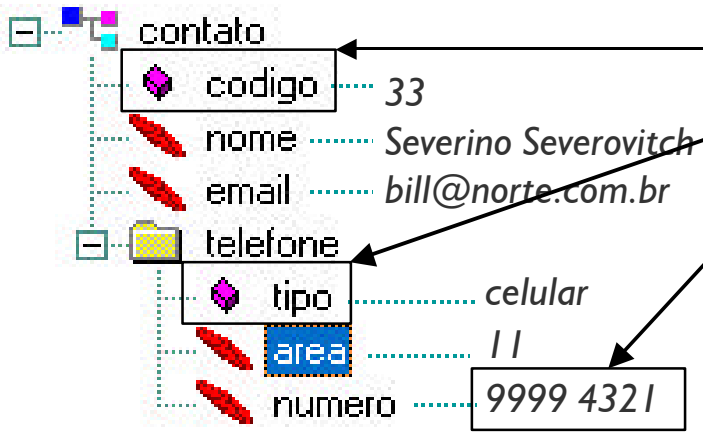
3. Árvore completa




```
doc.appendChild(carta)
```

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*

* pode também ser padrão de busca

```
<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

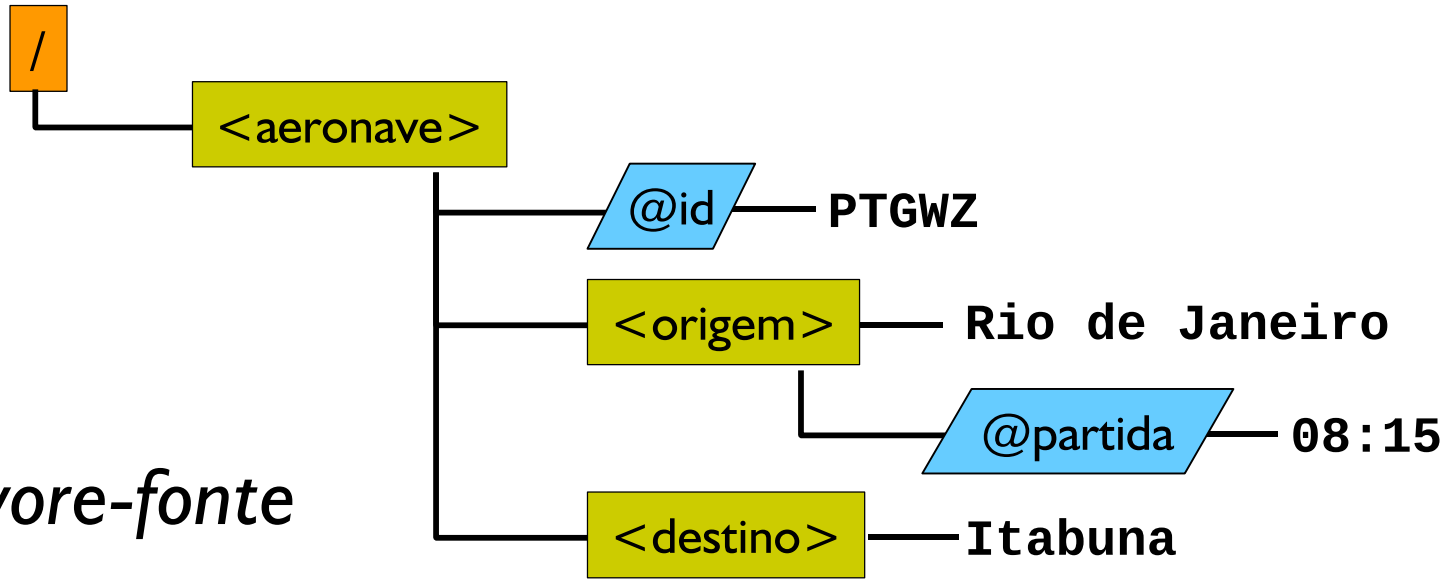
- ***XSL Transformations***
  - *Linguagem (XML) para criação de documentos que contêm regras de transformação para documentos XML*
  - *Documentos escritos em XSLT são chamados de* ***folhas de estilo*** *e contêm*
    - ***Elementos XSLT****: <template>, <if>, <foreach>, ...*
    - ***Expressões XPath*** *para localizar nós da árvore-fonte*
    - ***Texto ou XML*** *a ser gerado no documento-resultado*
  - *Processador XSLT*

XSLT: documento-fonte (I)

- *Considere o seguinte documento-fonte:*

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

- *Árvore-fonte*



XSLT: folha de estilos (2)

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

documento-fonte (em azul - XPath)

documento resultado (em preto)

```
<xsl:template match="aeronave">
    <p>A aeronave de prefixo
    <xsl:value-of select="@id" /> decolou
    de <xsl:value-of select="origem" /> às
    <xsl:value-of select="origem@partida" />
    tendo como destino o aeroporto de
    <xsl:value-of select="destino" />.</p>
</xsl:template>
```

elementos XSLT (em vermelho, com prefixo xsl)

- *Elementos XSLT geralmente são usados com um **prefixo** associado ao seu **namespace**: `<xsl:elemento>` para evitar conflitos com o documento-resultado.*

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
Itabuna.</p>
```

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

XLink, XPointer e XQuery

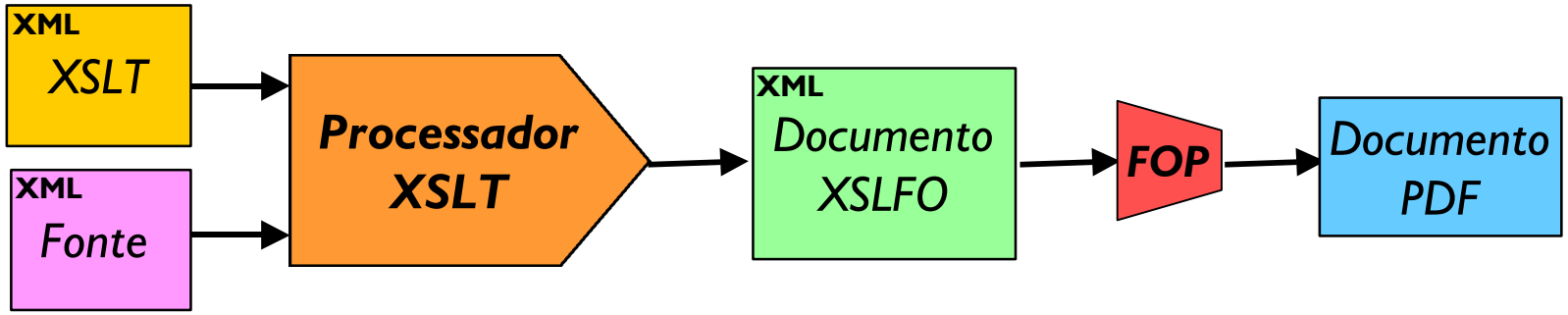
- **XLink**: *é uma especificação W3C que permite definir vínculos entre documentos XML*
  - *Funcionalidade mínima é igual ao <a href> do HTML*
  - *Funcionalidade estendida permite vínculos bidirecionais, arcos, vários níveis de semântica, etc.*
  - *É uma coleção de atributos, com namespace próprio, que podem ser usados em elementos de qualquer linguagem XML.*
- **XPointer**: *aponta para partes de documentos XML*
  - *Identificador (ID) colocado no destino, accessível através de fragmento de URL:* `xlink:href="#identificador"`
  - *Caminho resultante de expressão XPath:* `xpointer(/livro/id)`
- **XQuery**: *linguagem para pesquisar documentos XML*
  - *Exemplo:*

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

```
<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>
```

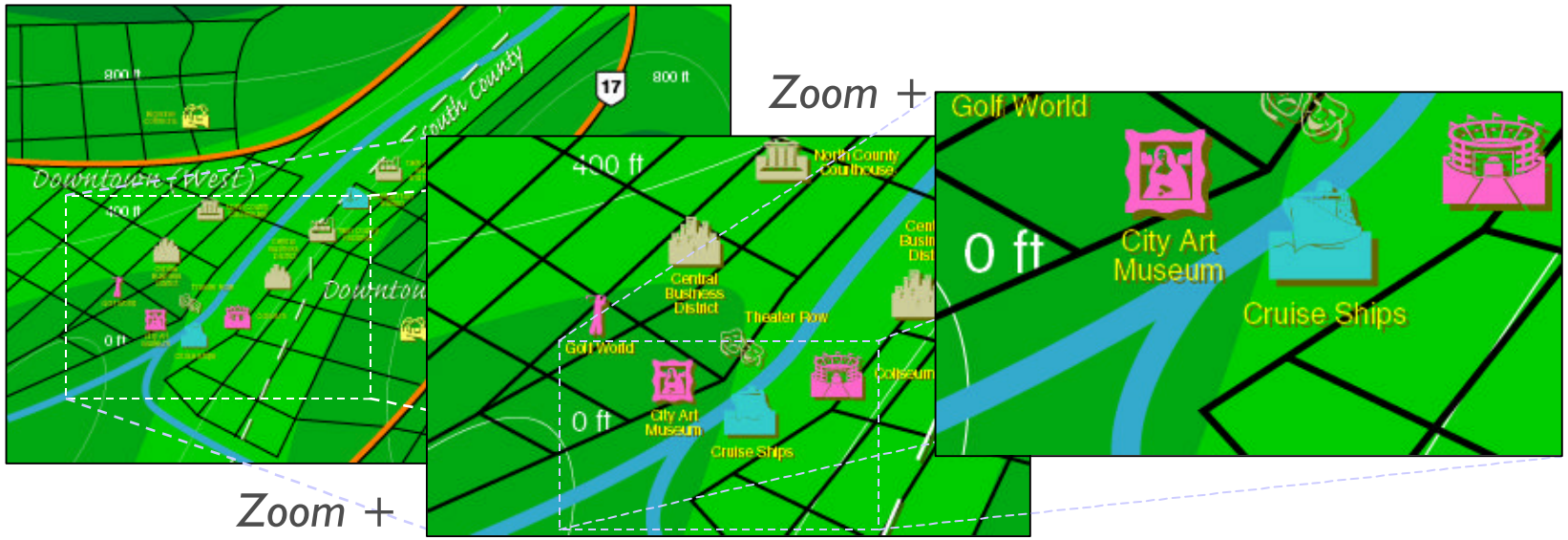
XHTML

- 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

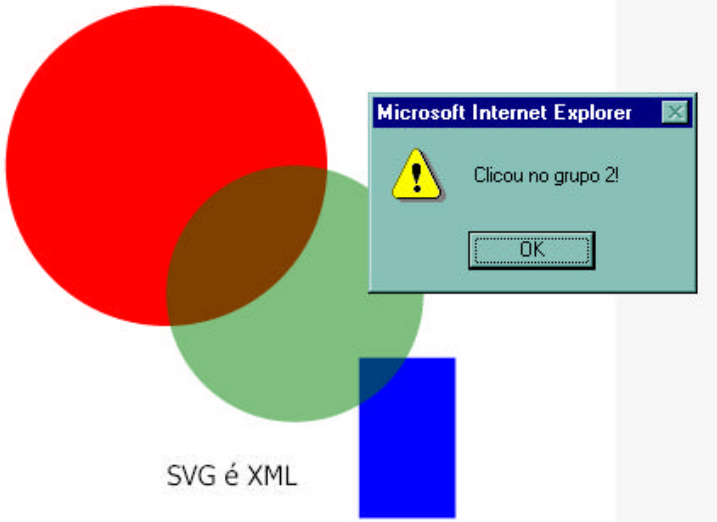
```
<svg>
  <circle style="fill: red" cx="3cm" cy="3cm" r="2.5cm" />
  <rect style="fill: blue" x="6cm" y="6cm"
        height="2.5cm" width="1.5cm" />
```

SVG

- *W3C **S**calable **V**ector **G**raphics*
  - *Gráficos vetoriais em XML*
  - *Plug-ins para principais browsers: concorre com Flash*
  - *Suporta animações, links, JavaScript, CSS*
  - *Produzido por ferramentas como Adobe Illustrator*
  - *Pode ser embutido no código XHTML e XSL-FO*



Exemplo de SVG



JavaScript

CSS

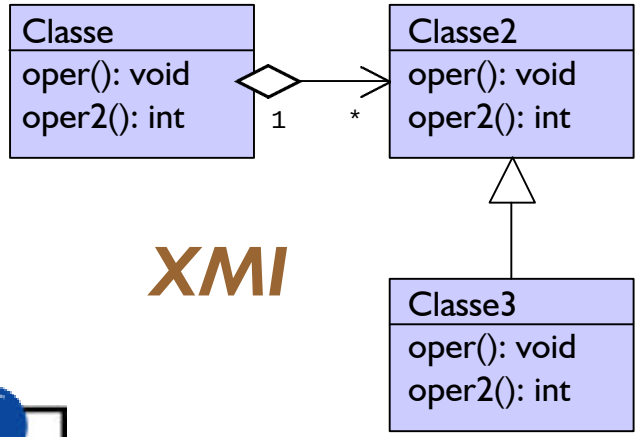
XLink

```
<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>
```

Algumas outras linguagens XML

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

MathML



XMI

WML

VoiceXML

Web Services

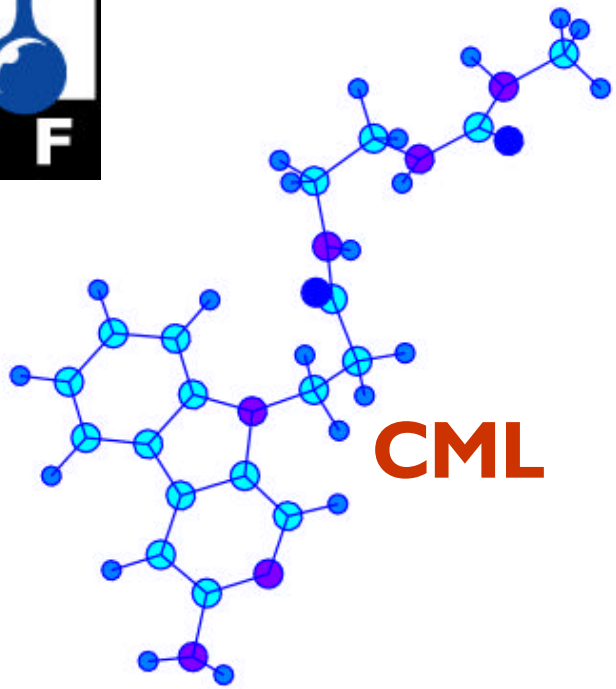
SOAP

WSDL

XML-RPC

UDDI

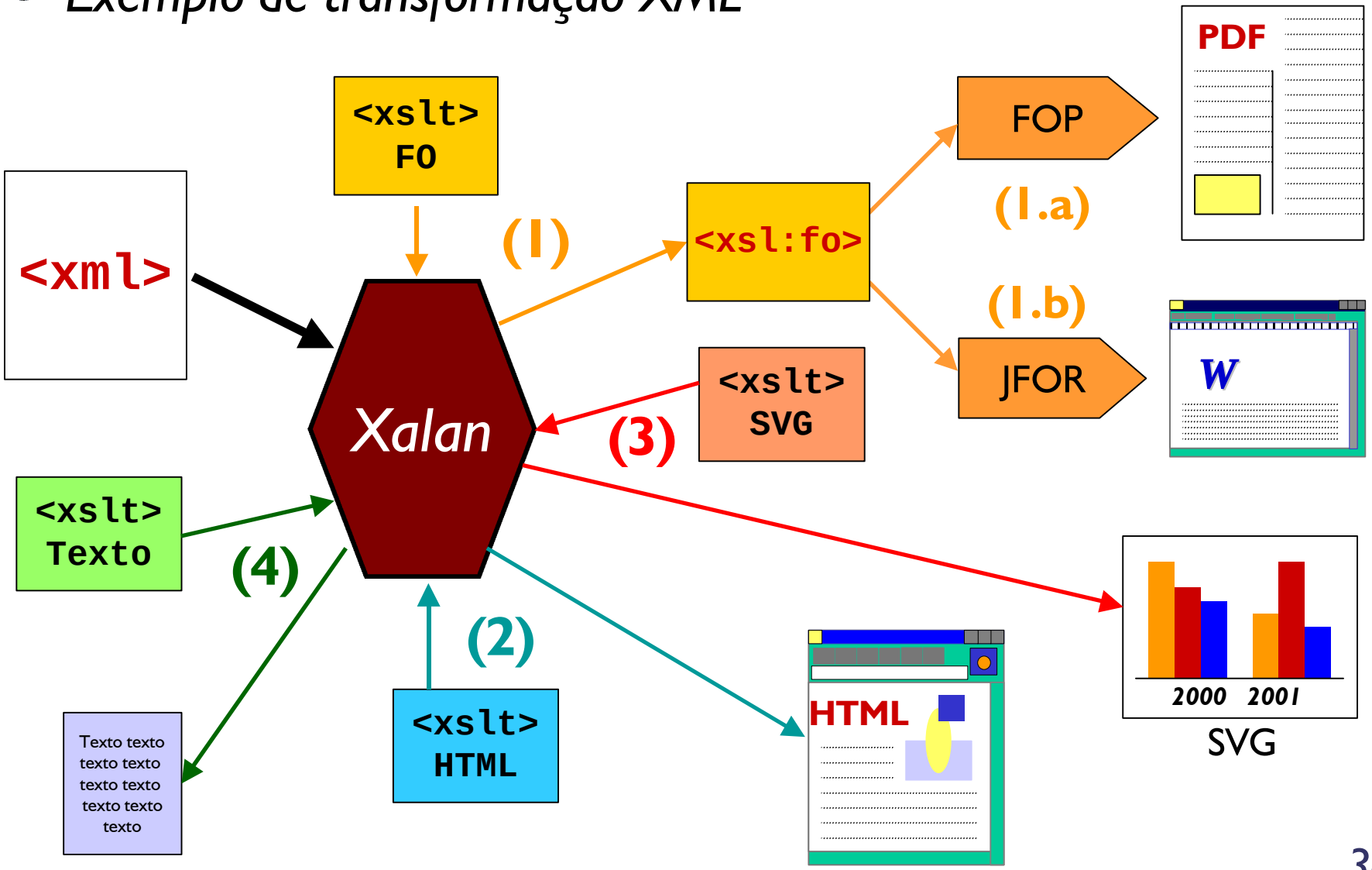
ebXML



CML

Demonstração

- *Exemplo de transformação XML*



Ferramentas

- Para programação
  - Parsers-validadores: Xerces, Crimson, MSXML 4.0
  - Validadores: MSV (Sun)
  - Transformadores XSL: TrAX, Xalan, Xt, Saxon
  - APIs: JDOM, JAX
  - Veja mais em xml.apache.org e www.alphaworks.ibm.com
- Para edição (de XML genérico)
  - XML Spy Suite
  - Framemaker / ArborText
  - JEdit com plug-ins para XML, XSLT e XPath
  - Veja mais em www.w3.org/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*

Fontes

[1] *World Wide Web Consortium (W3C). eXtensible Markup Language. http://www.w3.org/XML/. Ponto de partida e principal fonte sobre XML e suas tecnologias "satélite". Contém últimas especificações de XML, XPath, XSchema, XSLT, XSL-FO, XQuery, XLink, XPointer, SVG, XHTML, CSS.*

[2] *Eric Armstrong et al. Working with XML. Aborda DOM, SAX e XML com Java. http://java.sun.com/xml/jaxp/dist/1.1/docs/tutorial/index.html.*

[3] *Adobe. SVG Tutorial. http://www.adobe.com/svg/. Contém tutorial sobre SVG e links para o plug-in SVG da Adobe (Win/Mac).*

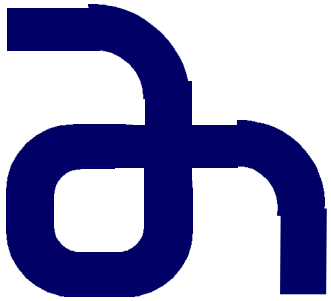
[4] *IBM Developerworks. http://www-106.ibm.com/developerworks/. Diversos tutoriais e artigos sobre XML, XSLT, DOM e SAX usando geralmente Java.*

[5] *Doug Tidwell. XSLT. O Reilly & Associates, 2001. Explora XSLT com aplicações práticas em Java.*

[6] *Elliotte Rusty Harold. XML Bible, Second Edition, 2001. Aborda todas as principais tecnologias W3C. 5 capítulos em http://cafeconleche.org/books/bible2/*

[7] *Erik T. Ray. Learning XML. O Reilly & Associates, 2001. Introdução ao XML e DTD, XSLT, XLink e XPointer (os dois últimos baseados em especificações draft).*

helder@ibpinet.net



www.argonavis.com.br/palestras/xml/