# XML Schema (xsD)

Prof. Dr. Alexandre L'Erario


# XML Schema

- XML pode sofrer atualizações e modificações

- Estas podem ocasionar erro de interpretação
  - Estrutura ou tipos modificados

- Exemplo:

  **<Primos> <primo>5</primo></Primos>**

  *Modificado para*

  **<Primos><primo id="5" /></Primos>**

  **<Primos> <primo>Cinco</primo></Primos>**

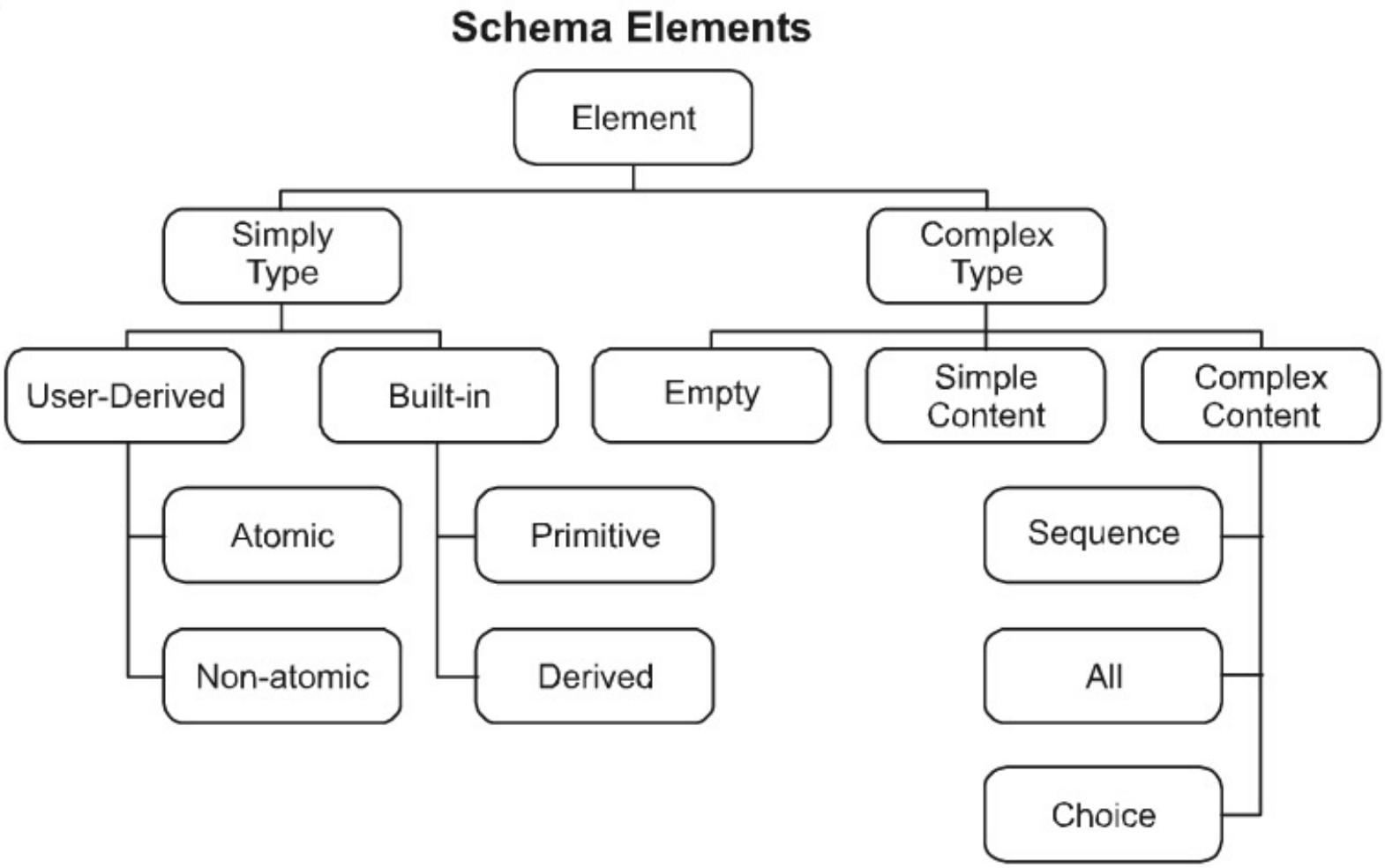
# XML Schema

- Descreve a estrutura de um documento XML
- Verifica se o documento é bem formado
- Identifica o tipo dos conteúdos
  - Apresenta um conjunto de tipos definidos
- Substitui o DTD (*Document Type Definition*)

# O que o xsD pode definir

- Atributos que podem aparecer no XML
- Quais são os elementos filhos
- A ordem e o número de elementos
- Tipos de dados aceitos
- Valores default ou requeridos
- A cardinalidade entre elementos

# Elementos descritos pelo Schema



# Tipos de elementos no schema

- Os elementos podem ser de tipo simples ou complexo
- Tipo Simples:
  - Podem conter apenas texto. Eles não podem ter elementos filhos ou atributos.
  - Todos os tipos integrados são tipos simples (por exemplo, xs:string).
  - Pode adotar um padrão específico (como e-mail, por exemplo)
  - Os tipos simples podem ser atômicos (por exemplo, strings e inteiros) ou não atômicos (por exemplo, listas).

- Tipo complexo
  - Podem conter elementos e atributos filho, bem como texto.
  - Por padrão, os elementos de tipo complexo contém elementos filhos.
  - Podem ser limitados a ter conteúdo simples, contendo apenas texto.
  - Podem ter atributos.
  - Podem ser limitados a não ter conteúdo (vazios).
  - Podem ter conteúdo misto - uma combinação de texto e elementos filho.

# Definição dos elementos

- Elemento simples:

  <xs:element name="xxx" type="yyy"/>

- Atributo:

  <xs:attribute name="xxx" type="yyy" />

- Tipos pré-definidos
  - xs:string
  - xs :decimal
  - xs:integer
  - xs:boolean
  - xs:date
  - xs:time

# Definição dos elementos

- Valor default
  - <xs:attribute name="lang" type="xs:string" fixed="EN"/>

- Valor requerido
  - <xs:attribute name="lang" type="xs:string" use="required"/>

# Restrições

| Restrição | Descrição |
|---|---|
| Enumeration | Define uma lista de valores aceitáveis |
| fractionDigits | Especifica o número máximo de casas decimais permitidas. Deve ser igual ou maior que zero |
| length | Especifica o número exato de caracteres ou itens de lista permitidos. Deve ser igual ou maior que zero |
| maxExclusive | Especifica os limites superiores para valores numéricos (o valor deve ser menor que este valor) |
| maxInclusive | Especifica os limites superiores para valores numéricos (o valor deve ser menor ou igual a este valor) |
| maxLength | Especifica o número máximo de caracteres ou itens da lista permitido. Deve ser igual ou maior que zero |
| minExclusive | Especifica os limites inferiores para valores numéricos (o valor deve ser maior que este valor) |
| minInclusive | Especifica os limites inferiores para valores numéricos (o valor deve ser maior ou igual a este valor) |
| minLength | Especifica o número mínimo de caracteres ou itens de lista permitidos. Deve ser igual ou maior que zero |
| pattern | Define a sequência exata de caracteres que são aceitáveis |
| totalDigits | Especifica o número exato de dígitos permitidos. Deve ser maior do que zero |
| whiteSpace | Especifica como o espaço em branco  é tratado |

# Restrição de valores

- Exemplo entre 0 e 120:

```
<xs:element name="age">
    <xs:simpleType>
        <xs:restriction base="xs:integer">
            <xs:minInclusive value="0"/>
            <xs:maxInclusive value="120"/>
        </xs:restriction>
    </xs: simpleType>
</xs:element>
```

# Restrição de valores

- Lista de valores aceitáveis: sedan, hatch, suv:

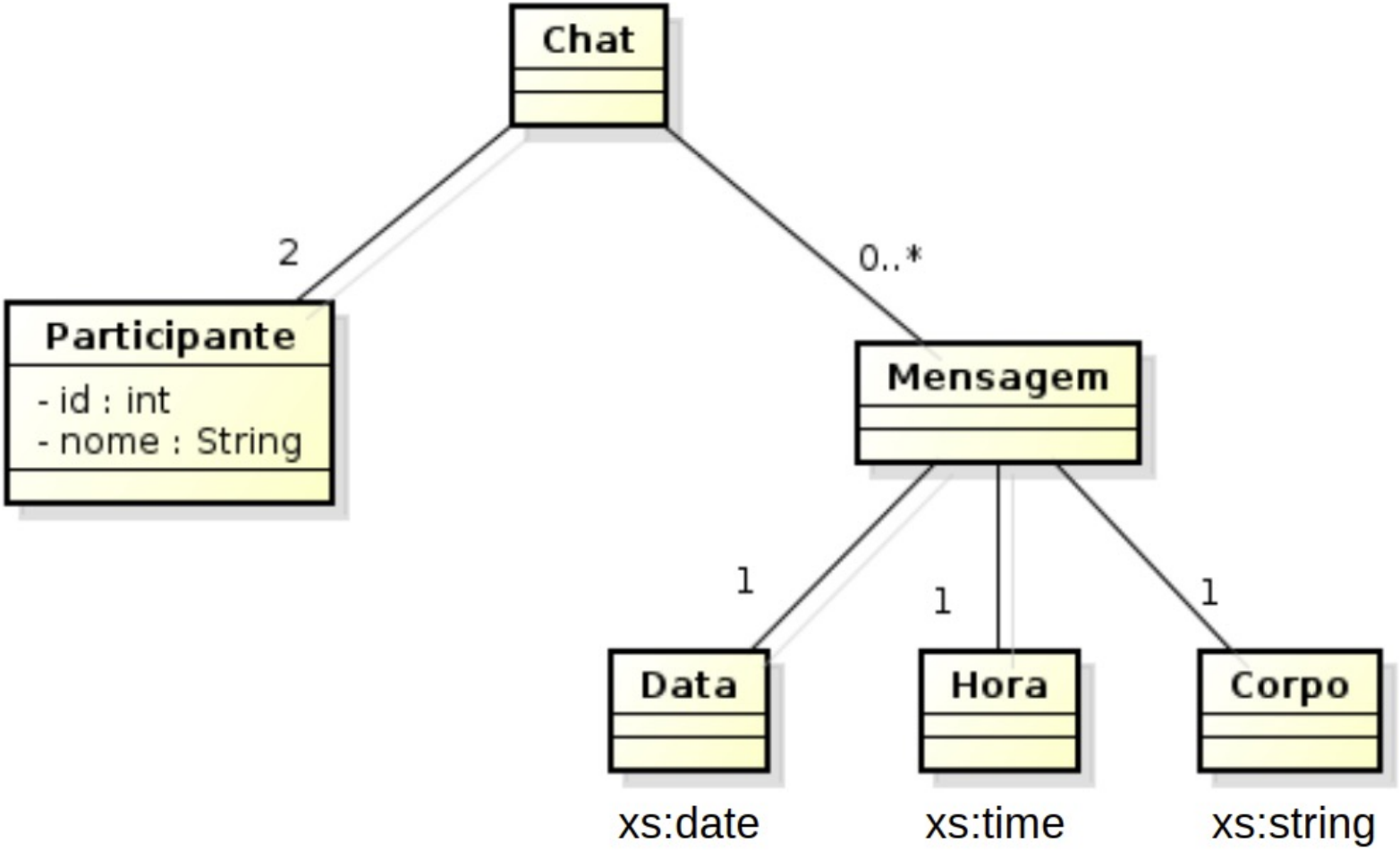
```
<xs:element name="car">
        <xs:simpleType>
                <xs:restriction base="xs:string">
                        <xs: enumeration value=”sedan"/>
                        <xs:enumeration value=”hatch"/>
                        <xs : enumeration value=”suv"/>
                </xs:restriction>
        </xs:simpleType>
</xs : element>
```

# Restrição de valores - Padrões

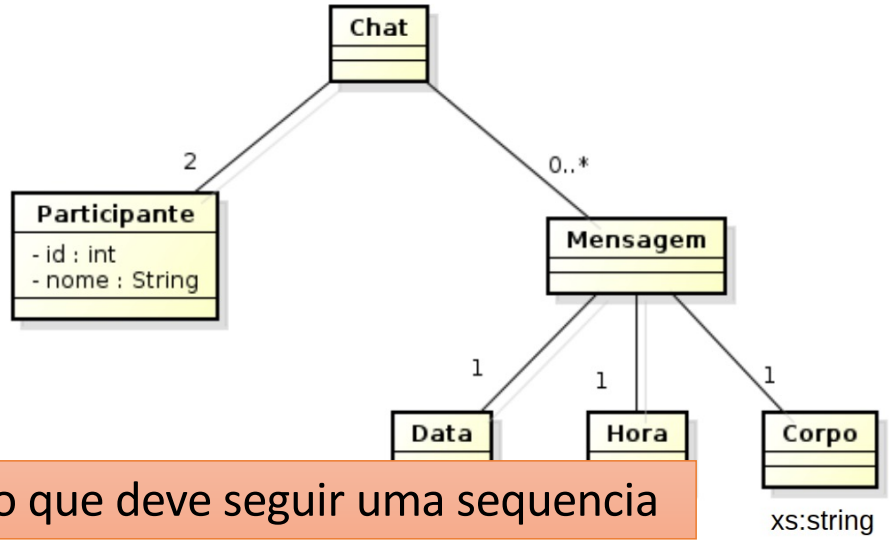
- Somente letras de a a z minúsculas:

```
<xs:element name="letter">
    <xs:simpleType>
        <xs:restriction base="xs:string">
            <xs:pattern value="[a-z]"/>
        </xs:restriction>
    </xs: simpleType>
</xs:element>
```

# Exemplo de um Schema



# Elemento Mensagem

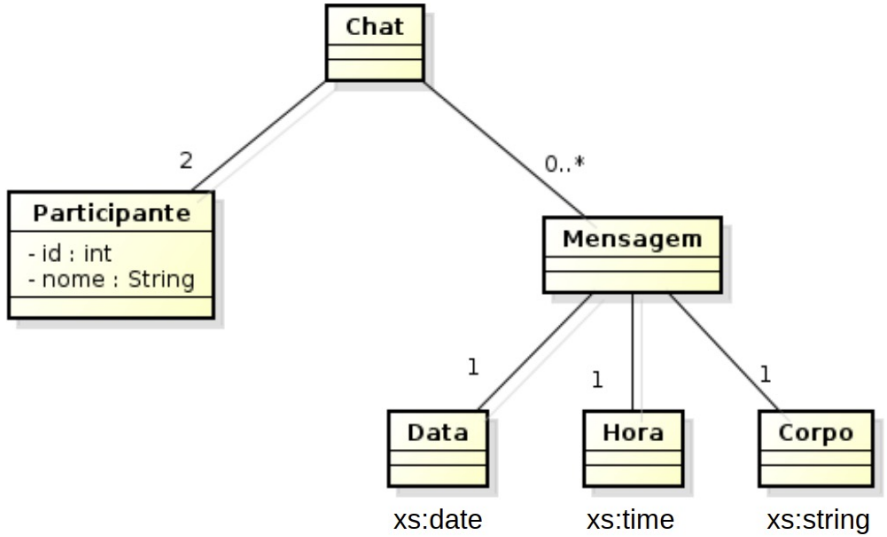



```
<xs:element name="Mensagem">
        <xs:complexType>
                <xs:sequence>
                        <xs:element name="Data" type="xs:date" minOccurs="1" maxOccurs="1" />
                        <xs:element name="Hora" type="xs:time" minOccurs="1" maxOccurs="1" />
                        <xs:element name="Corpo" type="xs:string" minOccurs="1" maxOccurs="1" />
                </xs:sequence>
        </xs:complexType>
</xs:element>
```

Tipo complexo que deve seguir uma sequencia

Elementos simples e seus tipos

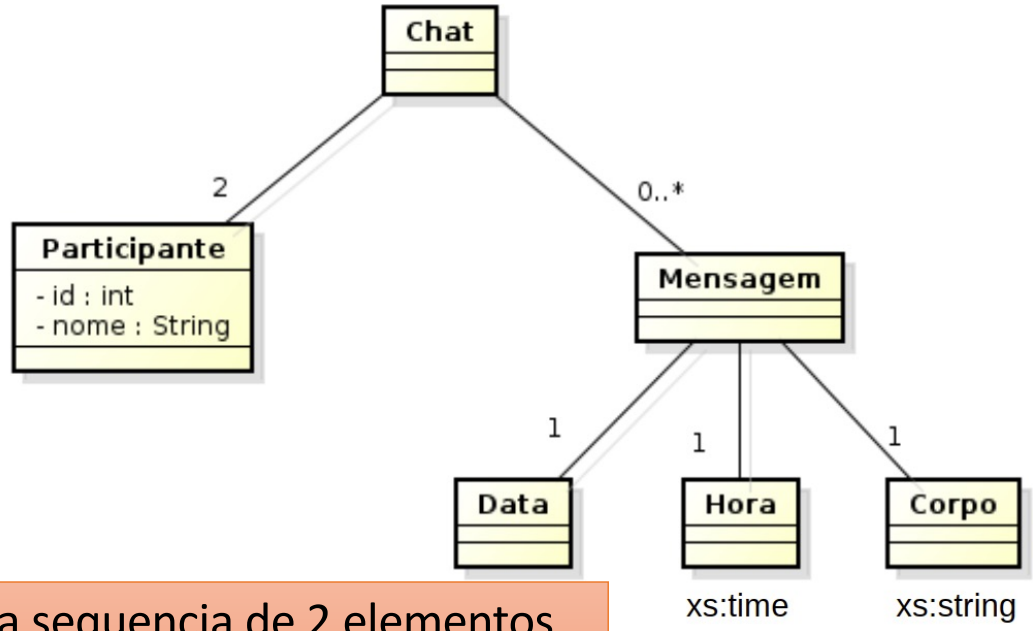
# Elemento Participante




```
<xs:element name="Participante">
        <xs:complexType>
                <xs:attribute name="id" type="xs:integer" use="required" />
                <xs:attribute name="nome" type="xs:string" use="required"/>
        </xs:complexType>
</xs:element>
```

Participante tem 2 atributos

# Elemento Chat




```
<xs:element name="Chat">
        <xs:complexType>
                <xs:sequence>
                        <xs:element ref="Participante" minOccurs="2" maxOccurs="2" />
                        <xs:element ref="Mensagem" minOccurs="0" maxOccurs="unbounded" />
                </xs:sequence>
        </xs:complexType>
</xs:element>
```

Chat é uma sequencia de 2 elementos

# XSD Final e XML Exemplo

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
  <xs:element name="Chat">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="Participante" maxOccurs="2" minOccurs="2"/>
        <xs:element name="Mensagem">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="Data" type="xs:date"/>
              <xs:element name="Hora" type="xs:time"/>
              <xs:element name="Corpo" type="xs:string"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:element name="Participante">
    <xs:complexType>
      <xs:attribute name="id" type="xs:int" use="optional"/>
      <xs:attribute name="nome" type="xs:string"/>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

```
<?xml version="1.0" encoding="utf-8"?>
<Chat>
<Participante id="123" nome="Alexandre" />
 <Participante id="321" nome="Laura" />
 <Mensagem>
  <Data>2021-01-24</Data>
  <Hora>21:32:52</Hora>
  <Corpo>Esta é uma mensagem</Corpo>
 </Mensagem>
</Chat>
```

# Vincular xsD e XML

```
<?xml version="1.0" encoding="UTF-8"?>
<Chat xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:noNamespaceSchemaLocation="file:/Chat.xsd">
   <Participante id="123" nome="Alexandre" />
   <Participante id="321" nome="Laura" />
   <Mensagem>
      <Data>2021-01-24</Data>
      <Hora>21:32:52</Hora>
      <Corpo>Esta é uma mensagem</Corpo>
   </Mensagem>
</Chat>
```

* Dentro do documento XML e/ou no aplicativo

# Considerações finais

- XSD é essencial para interoperabilidade
  - Padronização do XML
- Muitas linguagens/tecnologias possuem parser para validar XML com XSD