Esercitazione DTD-XSD

Paolo Papotti

http://papotti.dia.uniroma3.it/

- 18 Maggio 2009 -

Esercizio 1

- Descrivere uno schema per un elemento che descrive stringhe binarie
- binario contiene una serie di elementi uno o zero in qualsiasi ordine e con qualsiasi molteplicità

Soluzione DTD

- <!ELEMENT binario(zero*,uno*)*>
  <!ELEMENT zero(#PCDATA)>
  <!ELEMENT uno(#PCDATA)>

- <!ELEMENT binario(zero|uno)*>
  <!ELEMENT zero(#PCDATA)>
  <!ELEMENT uno(#PCDATA)>

Soluzione XML Schema

```
<xsd:element name="binario">
    <xsd:complexType>
        <xsd:choice minOccurs="0" maxOccurs="unbounded">
          <xsd:element name="zero" type="xsd:unsignedByte" />
          <xsd:element name="uno" type="xsd:unsignedByte" />
        </xsd:choice>
    </xsd:complexType>
</xsd:element>
```

Documenti istanza validi

```
<binario>
    <uno>1</uno>
    <zero>0</zero>
    <uno>1</uno>
    <uno>1</uno>
    <uno>1</uno>
    <zero>0</zero>
</binario>
```

```
<binario>
    <uno>0</uno>
    <zero>1</zero>
    <uno>0</uno>
    <uno>1</uno>
    <uno>0</uno>
    <zero>0</zero>
</binario>
```

Documenti istanza validi (per DTD)

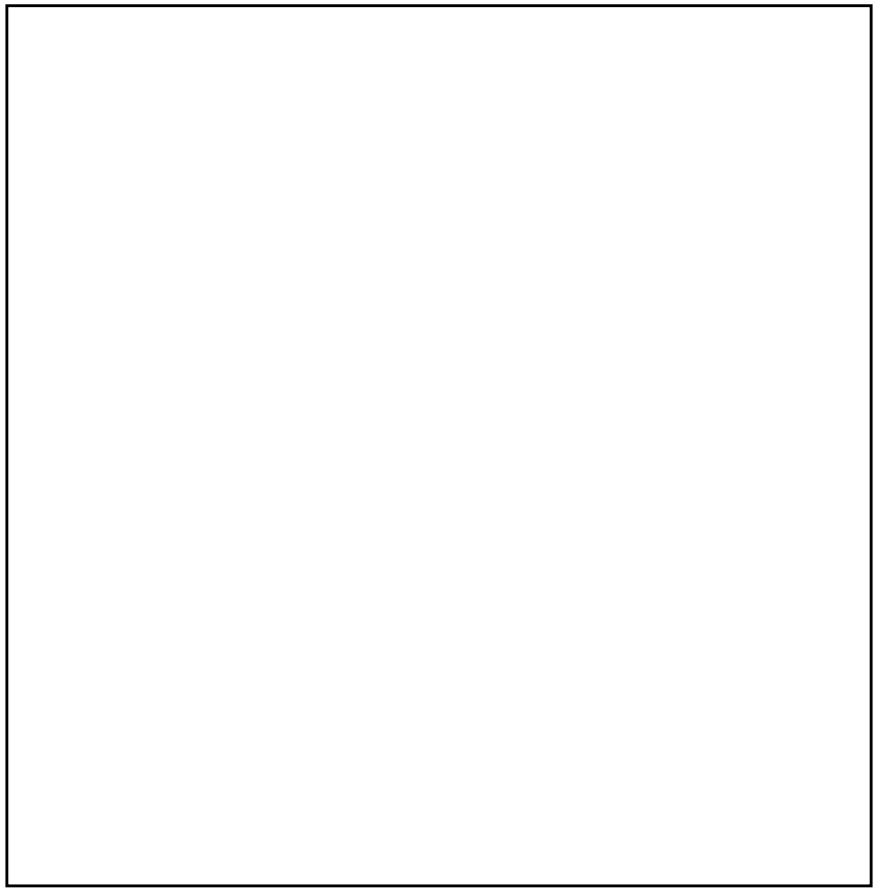
```
<binario>
    <uno>a</uno>
    <zero>0</zero>
    <uno>8</uno>
    <uno>5</uno>
    <uno>t</uno>
    <zero>0</zero>
</binario>
```

Soluzione XML Schema - raffinamento

```
<xsd:element name="binario">
    <xsd:complexType>
        <xsd:choice minOccurs="0" maxOccurs="unbounded">
            <xsd:element name="zero" type="xsd:unsignedByte"
                         fixed="0" />
            <xsd:element name="uno" type="xsd:unsignedByte"
                         fixed="1" />
        </xsd:choice>
    </xsd:complexType>
</xsd:element>
```

Documenti istanza validi

```
<binario>
    <uno>1</uno>
    <zero>0</zero>
    <uno>1</uno>
    <uno>1</uno>
    <uno>1</uno>
    <zero>0</zero>
</binario>
```



- Dato il seguente schema produrre un documento valido

```
<xsd:simpleType name="carattere">
      <xsd:restriction base="xsd:string">
            <xsd:pattern value="[A-Z]"/>
      </xsd:restriction>
</xsd:simpleType>
 <xsd:simpleType name="cifra">
      <xsd:restriction base="xsd:string">
            <xsd:pattern value="[0-9]"/>
      </xsd:restriction>
</xsd:simpleType>
<xsd:element name="car" type="carattere" />
<xsd:element name="num" type="cifra" />
<xsd:element name="esercizio" type="Tesercizio" />
<xsd:complexType name="Tesercizio">
      <xsd:sequence>
            <element ref="car" minOccurs="6" maxOccurs="6"/>
            <element ref="num" minOccurs="2"
                        maxOccurs="2"/>
            <element ref="car"/>
            <element ref="num" minOccurs="2"
                        maxOccurs="2"/>
             <element ref="car"/>
            <element ref="num" minOccurs="3"
                        maxOccurs="3"/>
            <element ref="car"/>
      </xsd:sequence>
</xsd:complexType>
```

soluzione

```
<esercizio>
    <car>X</car><car>X</car><car>X</car>
    <car>Y</car><car>Y</car><car>Y</car>
    <num>1</num> <num>1</num>
    <car>Z</car>
    <num>2</num> <num>2</num>
    <car>H</car>
    <num>3</num> <num>3</num> <num>3</num>
     <car>K</car>
</esercizio>
```

Design schemi XSD

```
<Book>
   <Title>Illusions</Title>
   <Author>Richard Bach</Author>
   <Cost>12.85</Cost>
</Book>
<Person>
   <Name>Richard Bach</Name>
</Person>
```

“Bambole russe”

```
<xsd:element name="Book">
    <xsd:complexType>
       <xsd:sequence>
          <xsd:element name="Title" type="xsd:string"/>
          <xsd:element name="Author" type="xsd:string"/>
          <xsd:element name="Cost">
            <xsd:simpleType>
              <xsd:restriction base="xsd:decimal"> <xsd:scale value="2"/>
    </xsd:restriction>
            </xsd:simpleType>
          </xsd:element>
        </xsd:sequence>
    </xsd:complexType>
</xsd:element>

<xsd:element name="Person">
    <xsd:complexType>
          <xsd:sequence>
                    <xsd:element name="Name" type="xsd:string"/>
          </xsd:sequence>
    </xsd:complexType>
</xsd:element>
```

“Tende alla veneziana”

```
<xsd:simpleType name="money">
    <xsd:restriction base="xsd:decimal">
          <xsd:scale value="2"/>
    </xsd:restriction>
</xsd:simpleType>
<xsd:complexType name="BookType">
    <xsd:sequence>
          <xsd:element name="Title" type="xsd:string"/>
          <xsd:element name="Author" type="xsd:string"/>
          <xsd:element name="Cost" type="money"/>
     </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="PersonType">
    <xsd:sequence>
          <xsd:element name="Name" type="xsd:string"/>
     </xsd:sequence>
</xsd:complexType>

<xsd:element name="Book" type="BookType"/>
<xsd:element name="Person" type="PersonType"/>
```

Esercizio 2

- Definire uno schema che permetta di validare il seguente documento:

```
<catenaMontuosa>
        <monte>
                <nome>Monte Bianco</nome>
                <regione>Valle d'Aosta</regione>
                <altezza unitaMisura="metri">4810</altezza>
        </monte>
</catenaMontuosa>
<catenaMontuosa>
        <monte>
                <nome>Gransasso</nome>
        </monte>
        …
</catenaMontuosa>
```

Soluzione DTD

<!ELEMENT catenaMontuosa(monte+)*>

<!ELEMENT monte(nome,regione?,altezza?)>

<!ELEMENT nome(#PCDATA)>
<!ELEMENT regione(#PCDATA)>
<!ELEMENT altezza(#PCDATA)>

<!ATTLIST altezza unitaMisura CDATA>

Che modello usare?

- Modello Bambole Russe:
  - Tutti i tipi sono anonimi
  - Tutti gli elementi sono locali
- Modello Fette di Salame:
  - Tutti i tipi sono anonimi
  - Tutti gli elementi sono globali
- Modello Giardino dell'Eden
  - Tutti i tipi (e gli attributi) sono nominali
  - Tutti gli elementi sono globali
- Modello Tende alla Veneziana
  - Tutti i tipi sono nominali
  - Tutti gli elementi sono locali

http://www.xfront.com/GlobalVersusLocal.html

Soluzione Modello Bambole Russe - 1

```
<xsd:schema xmlns… >
<xsd:element name="catenaMontuosa" maxOccurs="unbounded">
 <xsd:complexType><xsd:sequence>
    <xsd:element name="monte" maxOccurs="unbounded"/>
     <xsd:complexType><xsd:sequence>
          <xsd:element name="nome" type="xsd:string"/>
          <xsd:element name="regione" type="xsd:string"/>
          <xsd:element name="altezza">
                    …
          </xsd:element>
     </xsd:sequence></xsd:complexType>
    </xsd:element>
 </xsd:sequence></xsd:complexType>
</xsd:element>
```

- Tutti i tipi sono anonimi
- Tutti gli elementi sono locali

Soluzione Modello Bambole Russe - 2

```
...
        <xsd:element name="altezza">
          <xsd:complexType>
            <xsd:simpleContent>
                  <xsd:extension base="xsd:integer">
                    <xsd:attribute name="unitaMisura" type="xsd:string"/>
                  </xsd:extension>
            </xsd:simpleContent>
          </xsd:complexType>
        </xsd:element>
```

Soluzione XML Schema - 1 Modello Fette di Salame

```
<xsd:schema xmlns… >
<xsd:element name="catenaMontuosa">
    <xsd:complexType><xsd:sequence>
        <xsd:element ref="monte" maxOccurs="unbounded"/>
    </xsd:sequence></xsd:complexType>
</xsd:element>
<xsd:element name="monte" >
    <xsd:complexType><xsd:sequence>
        <xsd:element ref="nome"/>
        <xsd:element ref="regione"/>
        <xsd:element ref="altezza"/>
    </xsd:sequence></xsd:complexType>
</xsd:element>
```

- Tutti i tipi sono anonimi
- Tutti gli elementi sono globali

Soluzione XML Schema - 1

```
<xsd:schema xmlns… >
<xsd:element name="catenaMontuosa" maxOccurs="unbounded">
    <xsd:complexType><xsd:sequence>
        <xsd:element ref="monte" maxOccurs="unbounded"/>
    </xsd:sequence></xsd:complexType>
</xsd:element>
<xsd:element name="monte" >
    <xsd:complexType><xsd:sequence>
        <xsd:element ref="nome"/>
        <xsd:element ref="regione" minOccurs="0"/>
        <xsd:element ref="altezza" minOccurs="0"/>
    </xsd:sequence></xsd:complexType>
</xsd:element>
```

Soluzione XML Schema - 2

```
<xsd:element name="nome" type="xsd:string"/>
<xsd:element name="regione" type="xsd:string"/>
<xsd:element name="altezza" >
    <xsd:complexType>
      <xsd:simpleContent>
        <xsd:extension base="xsd:integer">
          <xsd:attribute name="unitaMisura" type="xsd:string"/>
        </xsd:extension>
      </xsd:simpleContent>
    </xsd:complexType>
</xsd:element>
```

Modello Fette di Salame

Esercizio 2 - raffinamento

- Definire uno schema che permetta di validare il seguente documento:

```
<catenaMontuosa>
    <monte>
        <nome>Monte Bianco</nome>
        <regione>Valle d'Aosta</regione>
        <altezza unitaMisura="metri">4810</altezza>
    </monte>
</catenaMontuosa>
...
```

e che permetta all'elemento catenaMontuosa di contenere altri elementi definiti in altri namespace.

Soluzione Esercizio 2 - raffinamento

```
<xsd:schema xmlns… >
<xsd:element name="catenaMontuosa" maxOccurs="unbounded">
 <xsd:complexType><xsd:sequence>
    <xsd:element name="monte" maxOccurs="unbounded"/>
     <xsd:complexType><xsd:sequence>
          <xsd:element name="nome" type="xsd:string"/>
          <xsd:element name="regione" type="xsd:string"/>
          <xsd:element name="altezza">
                    …
          </xsd:element>
     </xsd:sequence></xsd:complexType>
    </xsd:element>
    <xsd:any minOccurs="0" maxOccurs="unbounded"/>
 </xsd:sequence></xsd:complexType>
</xsd:element>
```

Finito?

- In generale si, ma io posso raffinare ulteriormente il mio schema:

```
<xsd:element name="regione" >
  <xsd:simpleType >
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="Valle d'Aosta"/>
      <xsd:enumeration value="Piemonte"/>
      <xsd:enumeration value="Lombardia"/>
      ...
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
```

Esercizio 3

- Definire uno schema che permetta di validare il seguente documento XML, utilizzando il modello Giardino dell'Eden:
  - Tutti i tipi (e gli attributi) sono nominali
  - Tutti gli elementi sono globali

```
<lettera xmlns=...>
    Gentile <cliente>Tal dei Tali</cliente>,
        la informiamo che i seguenti articoli da lei ordinati non sono più
    disponibili a magazzino:
    <ordine num="1234">
        <articolo><codice>1</codice>
                <descr>articolo 1</descr></articolo>
        <articolo><codice>5</codice>
                <descr>articolo 5</descr></articolo<
    </ordine>
    Cordiali saluti,
        <responsabile><tit>dr.<\tit> Mario Rossi</responsabile>
</lettera>
```

Esercizio 3 – soluzione - 1

```
<xsd:schema xmlns… >
<xsd:element name=“lettera” type=“Tlettera”/>
<xsd:element name=“cliente” type=“xsd:string”/>
<xsd:element name=“ordine” type=“Tordine”/>
<xsd:element name=“articolo” type=“Tarticolo” />
<xsd:element name=“codice” type=“xsd:string”/>
<xsd:element name=“descr” type=“xsd:string”/>
<xsd:element name=“responsabile” type=“Tresponsabile”/>
<xsd:element name=“tit” type=“xsd:string” />
<xsd:attribute name=“num” type=“xsd:integer”/>
<xsd:complexType name=“Tlettera” mixed=“true”>
    <xsd:sequence>
        <xsd:element ref=“cliente” /><xsd:element ref=“ordine”/>
        <xsd:element ref=“responsabile”/>
    </xsd:sequence>
</xsd:element>
```

Esercizio 3 – soluzione - 2

```
<xsd:complexType name="Tordine" >
    <xsd:sequence>
        <xsd:element ref="articolo" maxOccurs="unbounded"/>
    </xsd:sequence>
    <xsd:attribute ref="num"/>
</xsd:complexType>
<xsd:complexType name="Tarticolo">
    <xsd:sequence>
        <xsd:element ref="codice"/><xsd:element ref="descr"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="Tresponsabile" mixed="true" >
    <xsd:choice minOccurs="0"><xsd:element ref="tit"/></xsd:choice>
</xsd:complexType>
```

Esercizio 3 – soluzione - 3

```
<xsd:schema xmlns… >
<xsd:element name="lettera" type="Tlettera"/>
<xsd:complexType name="Tlettera" mixed="true">
    <xsd:sequence><xsd:element name="cliente" type="xsd:string"/>
        <xsd:element name="ordine" type="Tordine"/>
        <xsd:element name="responsabile" type="Tresponsabile"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="Tordine" >
    <xsd:sequence>
        <xsd:element name="articolo" type="Tarticolo"
                    maxOccurs="unbounded"/>
    </xsd:sequence>
    <xsd:attribute name="num" type="xsd:integer"/>
</xsd:complexType>
```

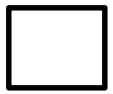
Esercizio 3 – soluzione - 4

```
<xsd:complexType name="Tarticolo">
    <xsd:sequence>
        <xsd:element name="codice" type="xsd:string"/>
        <xsd:element name="descr" type="xsd:string"/>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="Tresponsabile" mixed="true" >
    <xsd:choice minOccurs="0">
        <xsd:element name="tit" type="xsd:string" />
    </xsd:choice>
</xsd:complexType>
```

Che modello usare?

- Modello Bambole Russe:
  - Tutti i tipi sono anonimi
  - Tutti gli elementi sono locali
- Modello Fette di Salame:
  - Tutti i tipi sono anonimi
  - Tutti gli elementi sono globali
- Modello Giardino dell'Eden
  - Tutti i tipi (e gli attributi) sono nominali
  - Tutti gli elementi sono globali
- Modello Tende alla Veneziana
  - Tutti i tipi sono nominali
  - Tutti gli elementi sono locali

- I modelli Bambole Russe e Tende alla Veneziana permettono di scrivere meno codice
- Riutilizzo



  - Il modello Bambole russe non permette il riutilizzo
  - I modelli Fette di Salame e Giardino dell'Eden permettono di riutilizzare ogni elemento (o attributo) definito
  - Il modello Tende alla Veneziana permette il riutilizzo dei tipi e non degli elementi

Esercizio 5

- Definire un linguaggio XML che permetta di descrivere un documento di trasporto per ordini di libri. Ogni documento deve contenere i dati relativi all'acquirente (nome), l'indirizzo a cui spedire (nome, indirizzo, città, nazione) e di dati relativi ai libri ordinati (titolo, note[opzionale], quantità, prezzo)

Esercizio 5 - Documento XML

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<shiporder orderid="889923“ xmlns …>
    <orderperson>John Smith</orderperson>
    <shipto> <name>Ola Nordmann</name>
                <address>Langgt 23</address>
                <city>4000 Stavanger</city>
               <country>Norway</country>
    </shipto>
    <item> <title>Empire Burlesque</title>
          <note>Special Edition</note>
          <quantity>1</quantity> <price>10.90</price>
    </item>
    <item> <title>Hide your heart</title> <quantity>1</quantity>
             <price>9.90</price>
    </item>
</shiporder>
```

Esercizio 5 – Schema XML – Bambole Russe - 1

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="shiporder">
      <xs:complexType> <xs:sequence>
          <xs:element name="orderperson" type="xs:string"/>
          <xs:element name="shipto">
           <xs:complexType> <xs:sequence>
                    <xs:element name="name" type="xs:string"/>
                    <xs:element name="address" type="xs:string"/>
                    <xs:element name="city" type="xs:string"/>
                    <xs:element name="country" type="xs:string"/>
           </xs:sequence> </xs:complexType>
          </xs:element>
```

Esercizio 5 – Schema XML – Bambole Russe - 2

```
    <xs:element name="item" maxOccurs="unbounded">
      <xs:complexType> <xs:sequence>
          <xs:element name="title" type="xs:string"/>
          <xs:element name="note" type="xs:string"
                              minOccurs="0"/>
          <xs:element name="quantity"type="xs:positiveInteger"/>
          <xs:element name="price" type="xs:decimal"/>
      </xs:sequence> </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="orderid" type="xs:string" use="required"/>
 </xs:complexType>
</xs:element>
</xs:schema>
```

Esercizio 5 – Schema XML – Fette di Salame - 1

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="shiporder">
        <xs:complexType> <xs:sequence>
            <xs:element ref="orderperson"/>
            <xs:element ref="shipto"/>
            <xs:element ref="item" maxOccurs="unbounded"/>
        </xs:sequence> <xs:attribute ref="orderid" use="required"/>
        </xs:complexType>
    </xs:element>
    <xs:element name="orderperson" type="xs:string"/>
    <xs:element name="shipto">
        <xs:complexType> <xs:sequence>
            <xs:element ref="name"/>
            <xs:element ref="address"/>
            <xs:element ref="city"/>
            <xs:element ref="country"/>
        </xs:sequence> </xs:complexType>
    </xs:element>
```

- Tutti i tipi sono anonimi
- Tutti gli elementi sono globali

Esercizio 5 – Schema XML - Fette di Salame - 2

```
    <xs:element name="item">
        <xs:complexType> <xs:sequence>
            <xs:element ref="title"/>
            <xs:element ref="note" minOccurs="0"/>
            <xs:element ref="quantity"/>
            <xs:element ref="price"/>
        </xs:sequence> </xs:complexType>
    </xs:element>
    <xs:element name="name" type="xs:string"/>
    <xs:element name="address" type="xs:string"/>
    <xs:element name="city" type="xs:string"/>
    <xs:element name="country" type="xs:string"/>
    <xs:element name="title" type="xs:string"/>
    <xs:element name="note" type="xs:string"/>
    <xs:element name="quantity" type="xs:positiveInteger"/>
    <xs:element name="price" type="xs:decimal"/>
    <xs:attribute name="orderid" type="xs:string" />
</xs:schema>
```

Esercizio 5 – Schema XML – Giardino dell'Eden - 1

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="shiporder“ type=“Torder”/>
    <xs:element name="orderperson" type="xs:string"/>
    <xs:element name="shipto“  type=“Tshipto”/>
    <xs:element name="item“ type=“Titem”/>
    <xs:element name="name" type="xs:string"/>
    <xs:element name="address" type="xs:string"/>
    <xs:element name="city" type="xs:string"/>
    <xs:element name="country" type="xs:string"/>
    <xs:element name="title" type="xs:string"/>
    <xs:element name="note" type="xs:string"/>
    <xs:element name="quantity" type="xs:positiveInteger"/>
    <xs:element name="price" type="xs:decimal"/>
    <xs:attribute name="orderid" type="xs:string"/>
```

- Tutti i tipi (e gli attributi) sono nominali
- Tutti gli elementi sono globali

Esercizio 5 – Schema XML - Giardino dell'Eden - 2

```
<xs:complexType name="Torder">
 <xs:sequence>
      <xs:element ref="orderperson"/> <xs:element ref="shipto"/>
      <xs:element ref="item" maxOccurs="unbounded"/>
 </xs:sequence>
 <xs:attribute ref="orderid" use="required"/>
</xs:complexType>
<xs:complexType name="Tshipto"> <xs:sequence>
      <xs:element ref="name"/> <xs:element ref="address"/>
      <xs:element ref="city"/> <xs:element ref="country"/>
</xs:sequence></xs:complexType>
<xs:complexType name="Titem"> <xs:sequence>
      <xs:element ref="title"/> <xs:element ref="note" minOccurs="0"/>
      <xs:element ref="quantity"/><xs:element ref="price"/>
</xs:sequence> </xs:complexType>
</xs:schema>
```

Esercizio 5 – Schema XML – Tende alla Veneziana 1

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
    <xs:element name="shiporder" type="Torder"/>
    <xs:complexType name="Torder">
        <xs:sequence>
            <xs:element name="orderperson" type="xs:string"/>
            <xs:element name="shipto" type="Tshipto"/>
            <xs:element name="item" type="Titem" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="orderid" type="xs:string" use="required"/>
    </xs:complexType>
    <xs:complexType name="Tshipto"> <xs:sequence>
            <xs:element name="name" type="xs:string"/>
            <xs:element name="address" type="xs:string"/>
            <xs:element name="city" type="xs:string"/>
            <xs:element name="country" type="xs:string"/> </xs:sequence>
    </xs:complexType>
```

- Tutti i tipi sono nominali
- Tutti gli elementi sono locali

Esercizio 5 – Schema XML - Tende alla Veneziana 2

```
    <xs:complexType name=“Titem”>
      <xs:sequence>
           <xs:element name="title" type="xs:string"/>
           <xs:element name="note" type="xs:string“ minOccurs="0"/>
           <xs:element name="quantity" type="xs:positiveInteger"/>
           <xs:element name="price" type="xs:decimal"/>
       </xs:sequence>
    </xs:complexType>
</xs:schema>
```

Credits

- Questo materiale è basato su dispense ed esercizi di
  - Ombretta Gaggi @ Università di Padova
  - Alessio Pace, Valter Crescenzi, Paolo Merialdo @ Università Roma Tre