

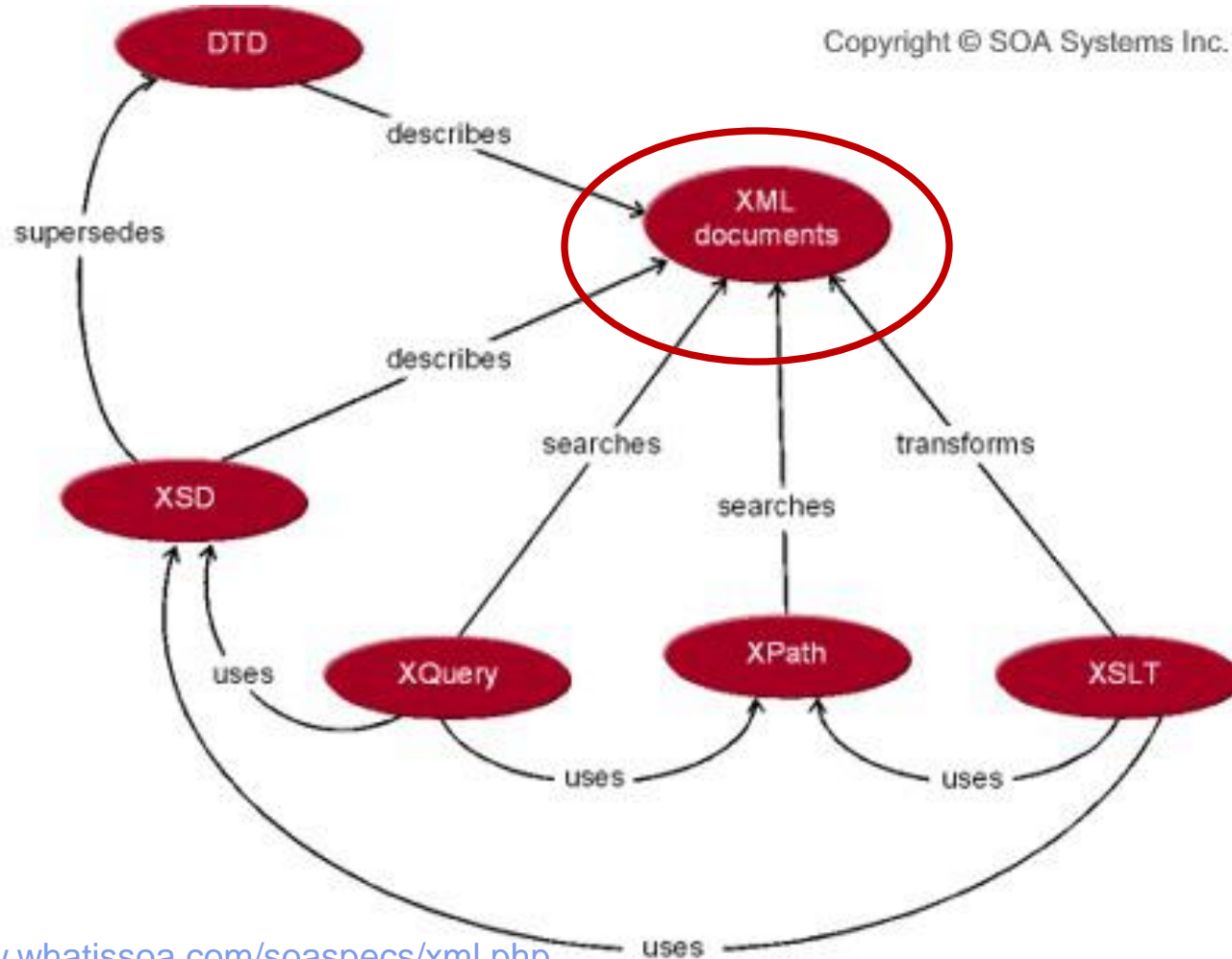
Introducción al middleware



Estándares XML



Estándares XML



- XML es un lenguaje de marcado
 - Utiliza “tags” para etiquetar, categorizar y organizar información

- XML no está limitado a un conjunto particular de tags
 - Es posible crear y utilizar “tags” de acuerdo a las necesidades

(Tidwell, 2002)



eXtensible Markup Language (XML)

```
<person>  
  <name>  
    <first_name>Alan</first_name>  
    <last_name>Turing</last_name>  
  </name>  
  <profession>computer scientist</profession>  
  <profession>mathematician</profession>  
  <profession>cryptographer</profession>  
</person>
```

<http://users.dimi.uniud.it/~massimo.franceschet/caffe-xml/xml/xml-elements.html>



eXtensible Markup Language (XML)

XML vs HTML

```
<p><b>Mrs. Mary McGoon</b>  
<br>  
1401 Main Street  
<br>  
Anytown, NC 34829</p>
```

Mrs. Mary McGoon
1401 Main Street
Anytown, NC 34829

```
<address>  
  <name>  
    <title>Mrs.</title>  
    <first-name>  
      Mary  
    </first-name>  
    <last-name>  
      McGoon  
    </last-name>  
  </name>  
  <street>  
    1401 Main Street  
  </street>  
  <city>Anytown</city>  
  <state>NC</state>  
  <postal-code>  
    34829  
  </postal-code>  
</address>
```

(Tidwell, 2002)



eXtensible Markup Language (XML)

Etiquetas, Elementos y Atributos

- Una **etiqueta** es el texto entre “<” y “>”
 - etiquetas de inicio: <name>
 - etiquetas de fin: </name>
- Un **elemento** es la etiqueta de inicio, la etiqueta de fin y lo contenido entre ambas
 - <name>.....</name>
- Un **atributo** es una pareja nombre-valor dentro de la etiqueta de inicio de un elemento:
 - <city state=“NC”>...

(Tidwell, 2002)



eXtensible Markup Language (XML)

Documentos Bien Formados y Válidos

- ❑ Documentos Bien Formados:
 - Son los que se adhieren a las reglas de sintaxis de XML

- ❑ Documentos Válidos:
 - Son los que se adhieren a las reglas de sintaxis de XML y a las reglas definidas en un DTD o Schema asociado



(Tidwell, 2002)

eXtensible Markup Language (XML)

Algunas reglas...

- ❑ Un documento XML debe estar contenido en un único elemento (root element)

- ❑ Los elementos no se pueden solapar
 - `<tag1> <tag2> </tag1> </tag2>`

- ❑ Las etiquetas de fin son requeridas

- ❑ Los elementos son “case sensitive”

(Tidwell, 2002)



eXtensible Markup Language (XML)

¿es un XML bien formado?

```
<?xml version="1.0" ?>
```

```
<cd>
```

```
<title>Empire Burlesque</title>
```

```
<artist>Bob Dylan</artist>
```

```
<country>USA</country>
```

```
</cd>
```

```
<cd>
```

```
<title>Hide your heart</title>
```

```
<artist>Bonnie Tyler</artist>
```

```
<country>UK</country>
```

```
</cd>
```



eXtensible Markup Language (XML)

¿es un XML bien formado?

```
<?xml version="1.0" ?>
```

```
<message>
```

```
  <to>Tove</to>
```

```
  <from>Jani</from>
```

```
  <subject>Reminder</subject>
```

```
  <body>Don't forget me this weekend!</body>
```

```
</message>
```



eXtensible Markup Language (XML)

¿es un XML bien formado?

```
<?xml version="1.0" ?>
<people>
  <person age=29>
    <name>Juan</name>
  </person>
  <person age=30>
    <name>Ana</Name>
  </person>
</people>
```



eXtensible Markup Language (XML)

Declaraciones XML

- Generalmente los documentos XML comienzan con una declaración que provee información sobre el documento
 - No es requerida en la v1.0
 - Es requerida en la v1.1

```
<?xml version="1.0" encoding="ISO-8859-1" standalone="no"?>
```

- standalone: especifica si el archivo puede ser procesado (o no) sin leer ningún otro archivo



eXtensible Markup Language (XML)

Comentarios y Entidades

□ Comentarios

<!-- un comentario XML -->

□ Entidades

- <!ENTITY uy "Uruguay"> / &uy;
- Entidades predefinidas:
 - < <
 - > >
 - " "
 - & &



eXtensible Markup Language (XML)

XML Namespaces

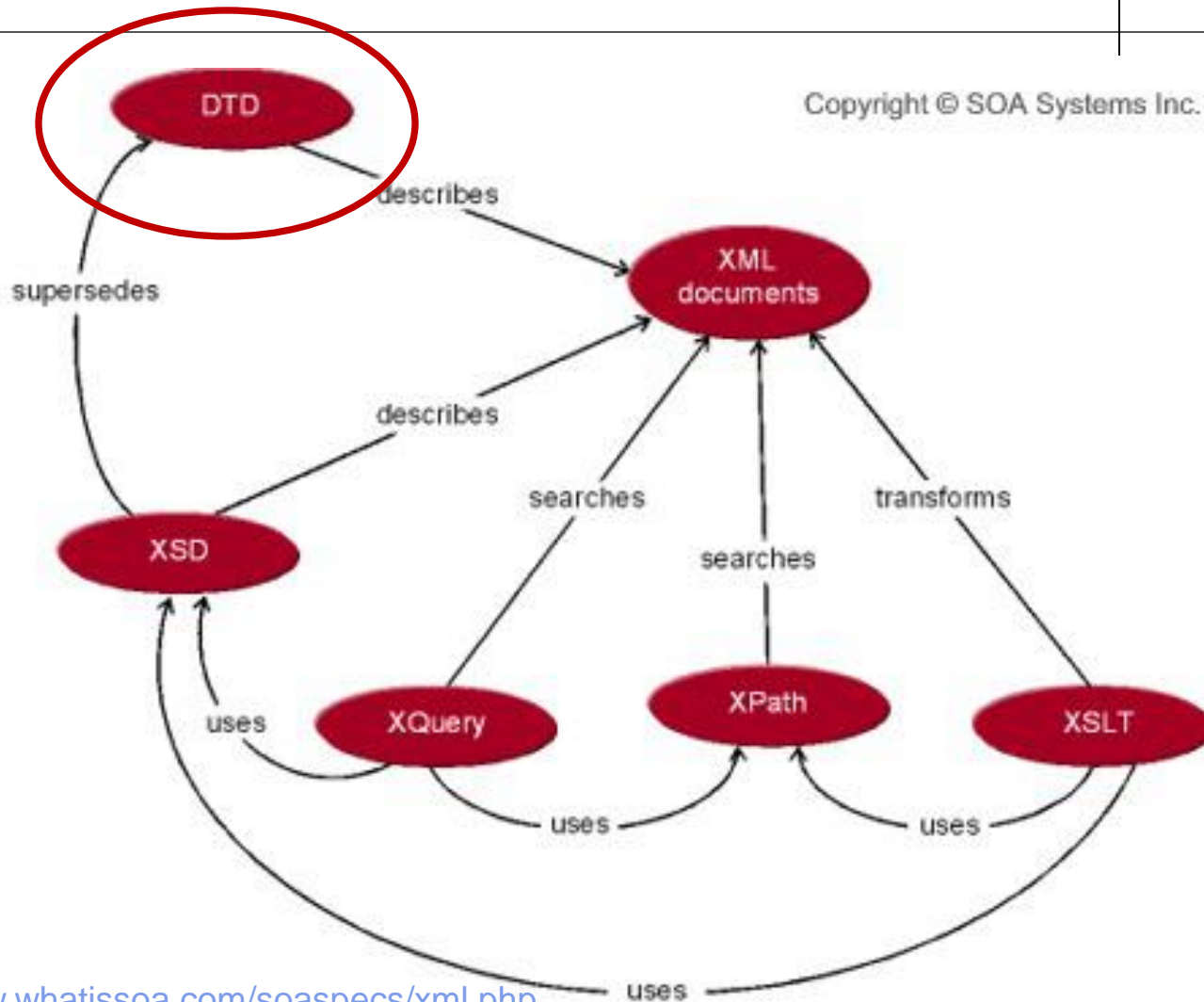
- Los XML Namespaces proveen un método para evitar conflictos en los nombres de los elementos

```
<customer_summary
  xmlns:addr="http://www.xyz.com/addresses/"
  xmlns:books="http://www.zyx.com/books/"
  xmlns:mortgage="http://www.yyz.com/title/"
>
<addr:name><title>Mrs.</title></addr:name>
<books:title>Lord of the Rings</books:title>
<mortgage:title>NC2948-388-1983</mortgage:title>
</customer_summary>
```

(Tidwell, 2002)



Estándares XML



- ❑ Los DTD describen de forma precisa
 - Qué elementos pueden aparecer en un documento XML
 - El orden en que pueden aparecer
 - Cuál es el contenido de un elemento
 - Cuáles son los atributos de un elemento
 - etc...

- ❑ En un DTD se puede especificar, por ejemplo, que:
 - Todo elemento *empleado* debe tener un atributo de nombre `numero_seguridad_social`.



Document Type Definition (DTD)

Elementos

□ Ejemplo

```
<!ELEMENT person (name, profession*)>
<!ELEMENT name (first_name, last_name)>
<!ELEMENT first_name (#PCDATA)>
<!ELEMENT last_name (#PCDATA)>
<!ELEMENT profession (#PCDATA)>
```

0 o más

□ Algunas Variaciones

```
<!ELEMENT person (name, profession+)>
<!ELEMENT name (firstname, m_name)>
<!ELEMENT name (firstname | last_name)>
```

```
<person>
  <name>
    <first_name>Alan</first_name>
    <last_name>Turing</last_name>
  </name>
  <profession>computer scientist</profession>
  <profession>mathematician</profession>
  <profession>cryptographer</profession>
</person>
```

1



Document Type Definition (DTD)

Atributos

- Ejemplo: `<!ELEMENT city (#PCDATA)>`
 - `<!ATTLIST city state CDATA #REQUIRED
postal-code CDATA #IMPLIED>`
 - `<city state="NC">...`
 - `<!ATTLIST city state CDATA (AZ|CA|NV) "CA">`
 - `<!ATTLIST city state CDATA #FIXED "CA">`



Document Type Definition (DTD)

Declaración Interna de un DTD

```
<?xml version="1.0"?>
```

```
<!DOCTYPE note [  
    <!ELEMENT note (to,from,heading,body)>  
    .....  
>
```

```
<note>
```

```
    <to>Tove</to>
```

```
    <from>Jani</from>
```

```
    <heading>Reminder</heading>
```

```
    <body>Don't forget me this weekend</body>
```

```
</note>
```



https://www.w3schools.com/xml/xml_dtd.asp

Document Type Definition (DTD)

Declaración Externa de un DTD

```
<?xml version="1.0"?>
```

```
<!DOCTYPE note SYSTEM "note.dtd">
```

```
<note>
```

```
  <to>Tove</to>
```

```
  <from>Jani</from>
```

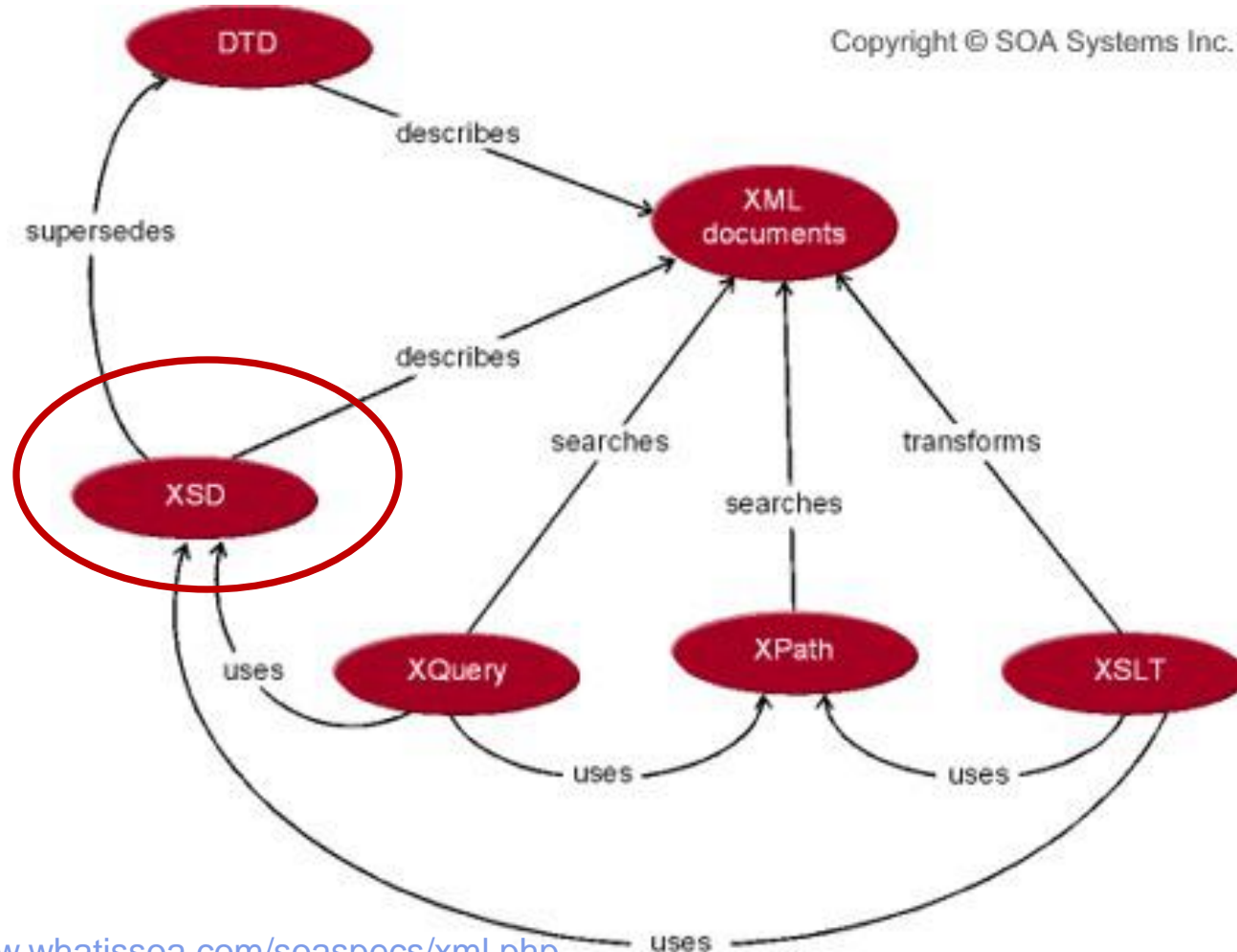
```
  <heading>Reminder</heading>
```

```
  <body>Don't forget me this weekend!</body>
```

```
</note>
```



Estándares XML



- ❑ Un XML Schema es un documento XML que contiene una descripción formal de cómo debe ser un documento XML válido.
- ❑ Algunas ventajas sobre los DTDs:
 - Utilizan sintaxis XML (son documentos XML)
 - Proveen soporte para tipos de datos (enteros, fechas, horas, etc)
 - Es extensible: se pueden definir nuevos tipos de datos
 - Tiene más poder de expresividad (ej, soporte para expresiones regulares)



□ XML Schema:

```
<xs:schema  
  xmlns:xs="http://www.w3.org/2001/XMLSchema">  
  <xs:element name="fullName" type="xs:string"/>  
</xs:schema>
```

□ Documento XML:

```
<fullName>Scott Means</fullName>
```

- xs:string
- xs:decimal
- xs:integer
- xs:boolean
- xs:date
- xs:time



□ Valores por defecto y fijos

- `<xs:element name="city" type="xs:string" default="Montevideo"/>`
- `<xs:element name="city" type="xs:string" fixed="Montevideo"/>`



□ Restricciones

```
<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>
```

http://www.w3schools.com/schema/schema_facets.asp



□ Elementos Complejos:

- Incluyen otros elementos y/o atributos

```
<xs:element name="persona">
```

```
<xs:complexType>
```

```
<xs:sequence>
```

```
<xs:element name="nombre" type="xs:string"/>
```

```
<xs:element name="apellido" type="xs:string"/>
```

```
</xs:sequence>
```

```
</xs:complexType>
```

```
</xs:element>
```



□ Elementos Complejos:

- Incluyen otros elementos y/o atributos

```
<xs:element name="persona" type="infoPersona"/>
```

```
<xs:complexType name="infoPersona">
```

```
  <xs:sequence>
```

```
    <xs:element name="nombre" type="xs:string"/>
```

```
    <xs:element name="apellido" type="xs:string"/>
```

```
  </xs:sequence>
```

```
</xs:complexType>
```



□ Elementos Complejos:

- Incluyen otros elementos y/o atributos

```
<xs:element name="persona">
```

```
<xs:complexType>
```

```
<xs:sequence>
```

```
<xs:element name="nombre" type="xs:string"/>
```

```
<xs:element name="apellido" type="xs:string"/>
```

```
</xs:sequence>
```

```
<xs:attribute name="CI" type="xs:string" use="required"/>
```

```
</xs:complexType>
```

```
</xs:element>
```



□ Indicadores de Orden

- all
 - Los elementos pueden aparecer en cualquier orden y sólo una vez
- sequence
 - Los elementos deben aparecer en el orden especificado
- choice
 - Sólo uno de los elementos puede aparecer



```
<xs:element name="person">  
  <xs:complexType>  
    <xs:all>  
      <xs:element name="firstname" type="xs:string"/>  
      <xs:element name="lastname" type="xs:string"/>  
    </xs:all>  
  </xs:complexType>  
</xs:element>
```



□ Indicadores de Ocurrencia (default 1)

- maxOccurs
- minOccurs

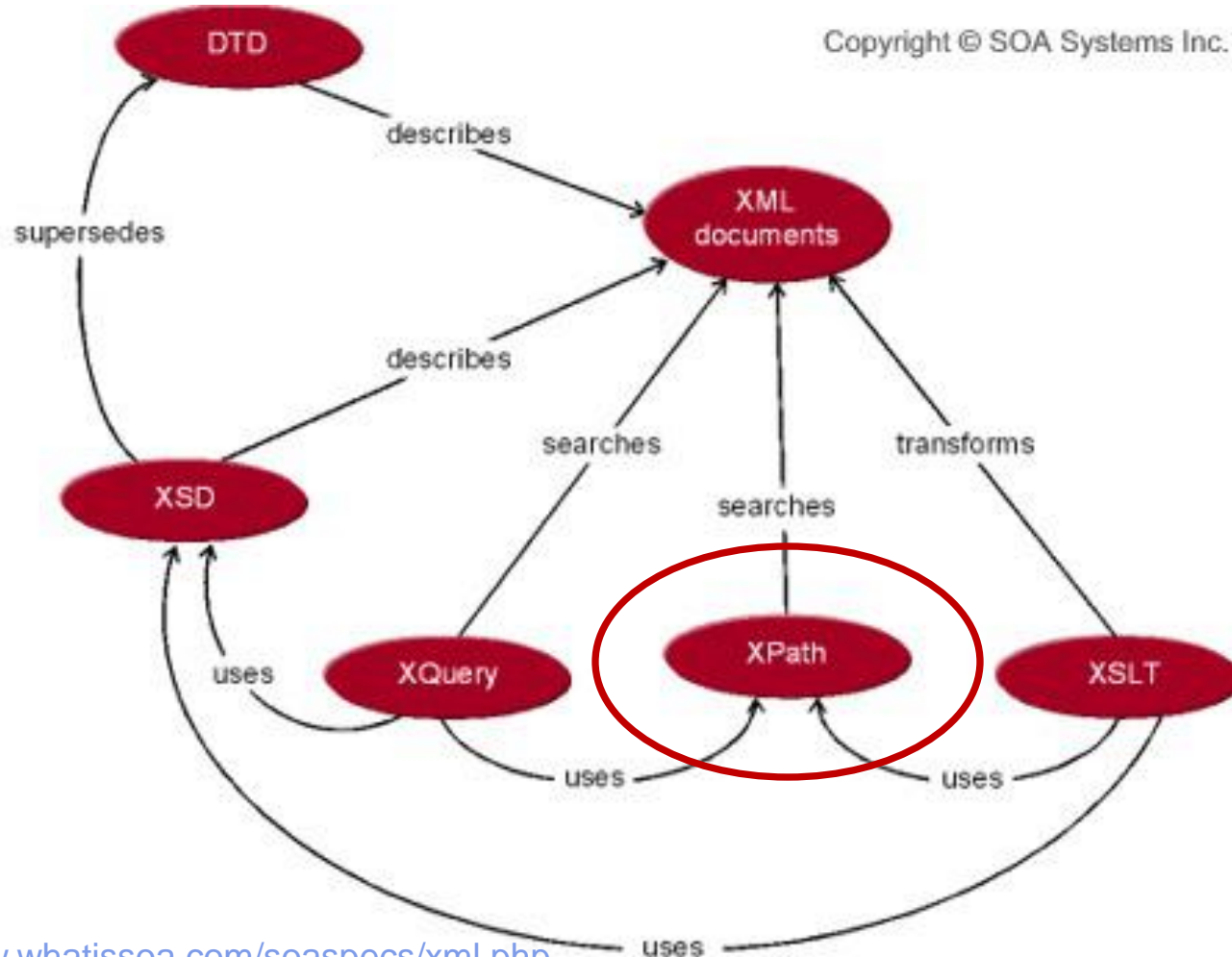
```
<xs:element name="person">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="full_name" type="xs:string"/>
      <xs:element name="child_name" type="xs:string"
        maxOccurs="10" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

maxOccurs="unbounded"

https://www.w3schools.com/xml/schema_complex_indicators.asp



Estándares XML



- ❑ XPath está basado en una representación arbórea de los documentos XML
- ❑ Provee la posibilidad de navegar por el árbol e ir seleccionado nodos del mismo según algún criterio definido.
- ❑ Permite realizar operaciones prefabricadas sencillas sobre los datos del documento, tales como operaciones matemáticas o lógicas.



XPath

Ejemplo

```
<people>
  <person id="342" >
    <name>
      <first_name>Juan</first_name>
      <last_name>Alvarez</last_name>
    </name>
  </person>
  <person id="4567" >
    <name>
      <first_name>Luis</first_name>
      <last_name>González</last_name>
    </name>
  </person>
</people>
```

Expresión:

/people/person/name/first_name

Resultado:

```
<first_name>Juan</first_name>
<first_name>Luis</first_name>
```



Expresión	Descripción
nodename	Selecciona todos los nodos de nombre nodename
/	Selecciona a partir de la raíz del documento
//	Selecciona nodos en cualquier lugar del documento
.	Selecciona el nodo actual
..	Selecciona el nodo padre del nodo actual
@	Selecciona atributos



- Ejemplos:
 - /people/person
 - //name
 - /people/person/@id
 - //@id



Expresión	Descripción
<code>/people/person[1]</code>	Selecciona el primer elemento person
<code>/people/person[last()]</code>	Selecciona el último elemento person
<code>/people/person[position()<3]</code>	Selecciona los dos primeros elementos person
<code>//person[@id='123']</code>	Selecciona todos los elementos person con id=123
<code>//person[age>50]</code>	Selecciona todos los elementos person con age>50
<code>//person[age>50]/name</code>	Selecciona los elementos name de todos los elementos person con age > 50



```
<year>1988</year>
</cd>
<cd id="3">
  <title>Greatest Hits</title>
  <artist>Dolly Parton</artist>
  <country>USA</country>
  <company>RCA</company>
  <price>9.90</price>
  <year>1982</year>
</cd>
<cd id="4">
  <title>Still got the blues</title>
  <artist>Garv Moore</artist>
```

XPath expression:

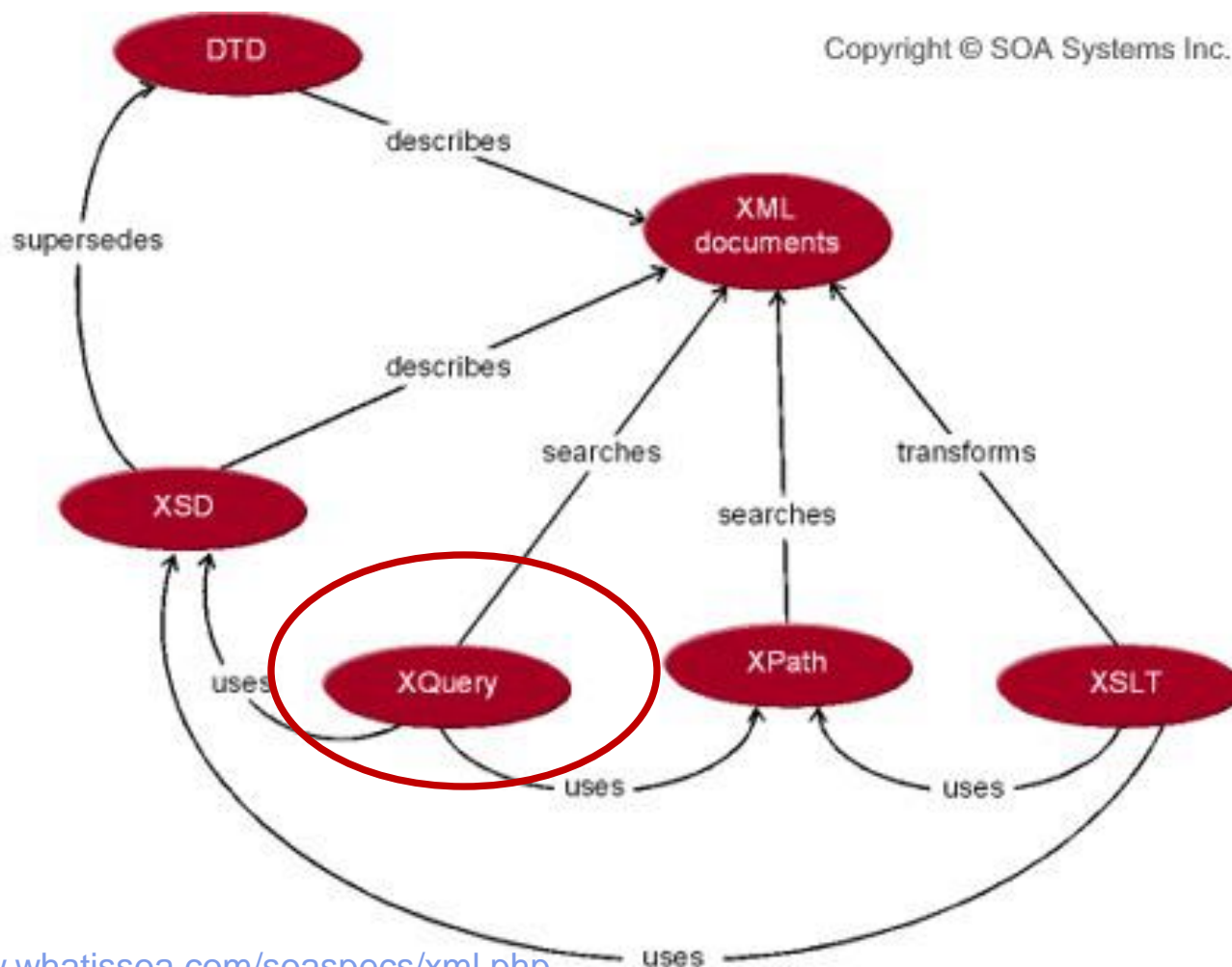
```
//cd[country="USA" and price<10]
```

Location: /catalog/cd[3]

- ▶ e cd id=3
 - e title
 - e artist
 - e country
 - e company
 - e price
 - e year
- ▶ e cd
- ▶ e cd
- ▶ e cd
- ▶ e cd



Estándares XML



- ❑ Lenguaje para consultar datos XML
 - "XQuery for XML is like SQL for databases"

- ❑ XQuery está construido sobre Xpath
 - `doc("people.xml")/people/person/name`

 - `doc("people.xml")/people/person[age<30]`



□ FLOWR (For, Let, Where, Order By, Return)

<results>

for \$person in doc (people.xml)/people/person

let \$lastname := \$person/name/last_name,

\$age := \$person/age

where \$age > 50

order by \$lastname

return

<personInfo>

{\$lastname}

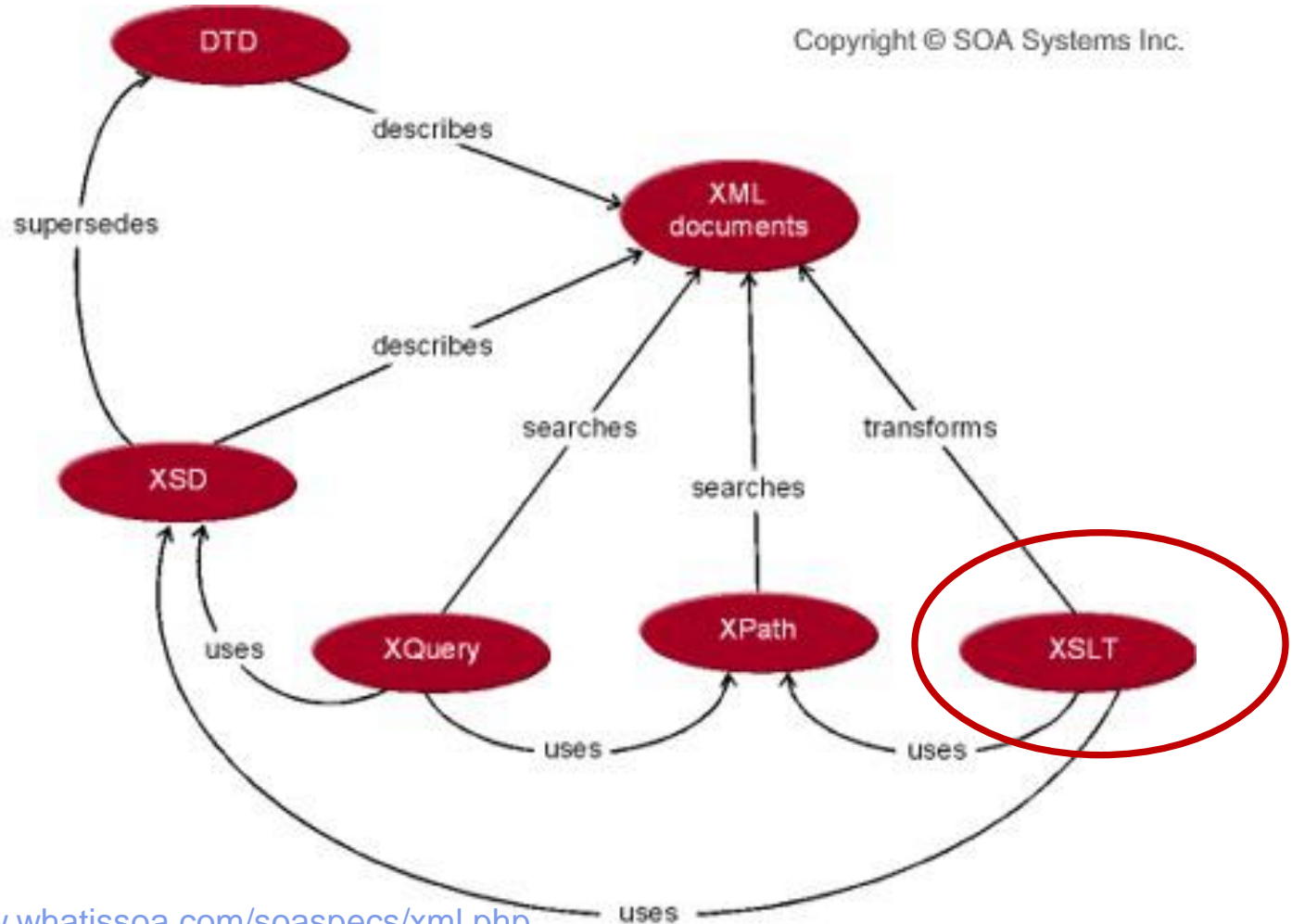
{\$age}

</personInfo>

</results>



Estándares XML



- ❑ eXtensible Stylesheet Language (XSL) está dividido en dos grandes partes:
 - XSL Transformations (XSLT)
 - XSL Formatting Objects (XSL-FO)

- ❑ XSLT es una aplicación del lenguaje XML para especificar reglas que permiten transformar un documento XML en otro.



- ❑ XSLT puede utilizarse para comunicar aplicaciones que manejan datos con diferente formato, vía una relación entre esquemas XML.
- ❑ XSLT utiliza expresiones XPath para identificar los nodos que siguen determinado patrón.



XSLT

Ejemplo

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<catalog>
  <cd>
    <title>Empire Burlesque</title>
    <artist>Bob Dylan</artist>
    <country>USA</country>
    <company>Company</company>
    <price>10.90</price>
    <year>1985</year>
  </cd>
  <cd>
    ....
  </cd>
</catalog>
```

My CD Collection

Title	Artist
Empire Burlesque	Bob Dylan
Hide your heart	Bonnie Tyler
Greatest Hits	Dolly Parton
Still got the blues	Gary Moore



XSLT

Ejemplo

```
<?xml version="1.0" encoding="ISO-8859-1"?>  
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
```

```
<xsl:template match="/">
```

```
<html>
```

```
<body>
```

```
<h2>My CD Collection</h2>
```

```
<table border="1">
```

```
<tr bgcolor="#9acd32">
```

```
<th>Title</th>
```

```
<th>Artist</th>
```

```
</tr>
```

```
<xsl:for-each select="catalog/cd">
```

```
<tr>
```

```
<td><xsl:value-of select="title"/></td>
```

```
<td><xsl:value-of select="artist"/></td>
```

```
</tr>
```

```
</xsl:for-each>
```

```
</table>
```

```
</body>
```

```
</html>
```

```
</xsl:template>
```

```
</xsl:stylesheet>
```

http://www.w3schools.com/xsl/xsl_transformation.asp



XSLT

Ejemplo

```
<xsl:for-each select="catalog/cd">
  <xsl:if test="price > 10">
    <tr>
      <td><xsl:value-of select="title"/></td>
      <td><xsl:value-of select="artist"/></td>
    </tr>
  </xsl:if>
</xsl:for-each>
```

http://www.w3schools.com/xsl/xsl_if.asp

```
<xsl:for-each select="catalog/cd">
  <xsl:sort select="artist"/>
  <tr>
    <td><xsl:value-of select="title"/></td>
    <td><xsl:value-of select="artist"/></td>
  </tr>
</xsl:for-each>
```

https://www.w3schools.com/xml/xsl_sort.asp



XSLT

Ejemplo

```
<xsl:for-each select="catalog/cd">
<tr>
  <td><xsl:value-of select="title"/></td>
  <xsl:choose>
    <xsl:when test="price > 10">
      <td bgcolor="#ff00ff">
        <xsl:value-of select="artist"/></td>
      </xsl:when>
      <xsl:otherwise>
        <td><xsl:value-of select="artist"/></td>
      </xsl:otherwise>
    </xsl:choose>
  </tr>
</xsl:for-each>
```



Preguntas



Referencias

- Tidwell, D. (2002, August 7). Introduction to XML. <http://www.ibm.com/developerworks/xml/tutorials/xmlintro/>

