Hojas de Estilos XSLT en el aula

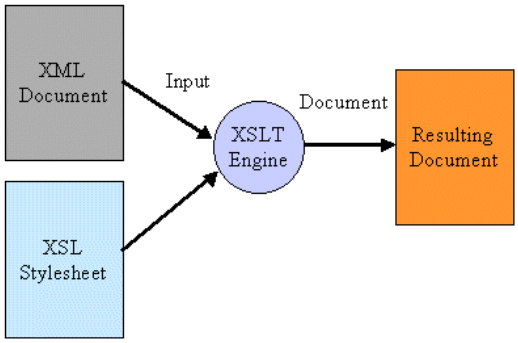
Nieves Carralero Colmenar
I.E.S Ramón y Cajal. Albacete
ncarralero@jccm.es

Resumen

Según la Orden EDU/2887/2010, de 2 de noviembre, por la que se establece el currículo del ciclo formativo de Grado Superior correspondiente al título de Técnico Superior en Desarrollo de Aplicaciones Web, se define un módulo con título: “Módulo Profesional: Lenguajes de marcas y sistemas de gestión de información”. En él se pretende hacer una aproximación al trabajo con XSLT como pieza clave en el desarrollo de aplicaciones Web.

1. INTRODUCCIÓN A XSLT

XSLT es la más importante parte del estándar XSL. Esta parte es usada para transformar un documento XML en otro documento XML o de otro tipo que sea reconocible por un browser, como HTML o XHTML. Normalmente es usado en aplicaciones Web para transformar XML en (X)HTML y dar formato personalizado a los datos. XSLT puede añadir nuevos elementos a un fichero de salida, o eliminar elementos que no sean necesarios. Puede cambiar y ordenar los elementos e incluso tomar decisiones sobre qué elementos se deben visualizar. Una imagen que esquematiza los elementos participantes es la siguiente:



XSLT usa XPath [XML6] para definir el mapeo entre las transformaciones. En resumen, XPath define que partes de un documento que encajan (match) con una o más plantillas (templates). Cuando una coincidencia es encontrada, XSLT transforma la parte coincidente en el documento fuente en el documento resultado. Las partes del documento fuente que no encajan con una plantilla no son incluidos en el documento resultado.

Declarando una hoja de Estilos (Style sheet)

El elemento raíz que declara que un documento es una hoja de estilos es <xsl:stylesheet> or <xsl:transform>. Ambos son sinónimos y pueden ser usados. El camino correcto para declarar una hoja de estilos según la recomendación del W3C XSLT es la siguiente:

```
<xsl:stylesheet
version="1.0"xmlns:xsl="http://www.w3.org/1999/XSL/Transform">or
:

<xsl:transform
version="1.0"xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
```

xmlns:xsl="http://www.w3.org/1999/XSL/Transform" identifica el espacio de nombres de la recomendación W3C XSL. Si se usa este espacio de nombres también se debe incluir el atributo version="1.0"

Las siguientes prácticas se realizarán con el ejemplo Books.xml que se muestra a continuación y que ha sido usado en previas publicaciones [XML2][XML3][XML4][XML5]:

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="xslbook.xsl"?>
<Books>
    <Book Pages="1088">
                <Author>Delaney, Kalen</Author>
                <Title>Inside Microsoft SQL Server 2000</Title>
        <Publisher>Microsoft Press</Publisher>
    </Book>
    <Book Pages="997">
        <Author>Burton, Kevin</Author>
        <Title>.NET Common Language Runtime</Title>
        <Publisher>Sams</Publisher>
    </Book>
    <Book Pages="392">
        <Author>Cooper, James W.</Author>
        <Title>C# Design Patterns</Title>
        <Publisher>Addison Wesley</Publisher>
    </Book>
</Books>
```

Observar que con el atributo href="xslbook.xsl" se hace referencia a que sobre ese documento XML se aplicará una plantilla llamada xslbook.xsl. Los ejemplos

que se ponen a continuación se llamarán xslbook.xsl para poder ser aplicados sobre este documento XML.

2. XSLT EJEMPLO INICIAL.

La siguiente figura muestra un ejemplo (xslbookp1.xsl) de documento XSL para convertir un documento XML (Books.xml) en un documento HTML que visualiza los datos como una tabla.

```
<?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>Librería Libro.NET</h2>
                <table border="1">
                <tr bgcolor="#9acd32">
                        <th align="left">Titulo</th>
                        <th align="left">Autor</th>
                </tr>
                <xsl:for-each select="Books/Book">
                <tr>
                        <td><xsl:value-of select="Title"/></td>
                        <td><xsl:value-of select="Author"/></td>
                </tr>
                </xsl:for-each>
                </table>
        </body>
 </html>
</xsl:template></xsl:stylesheet>
```

El resultado se aplicar esta plantilla sobre el Book.XML es la siguiente interpretación de HTML:

Librería Libro.NET

Titulo	Autor
Inside Microsoft SQL Server 2000	Delaney, Kalen
.NET Common Language Runtime	Burton, Kevin
C# Design Patterns	Cooper, James W.

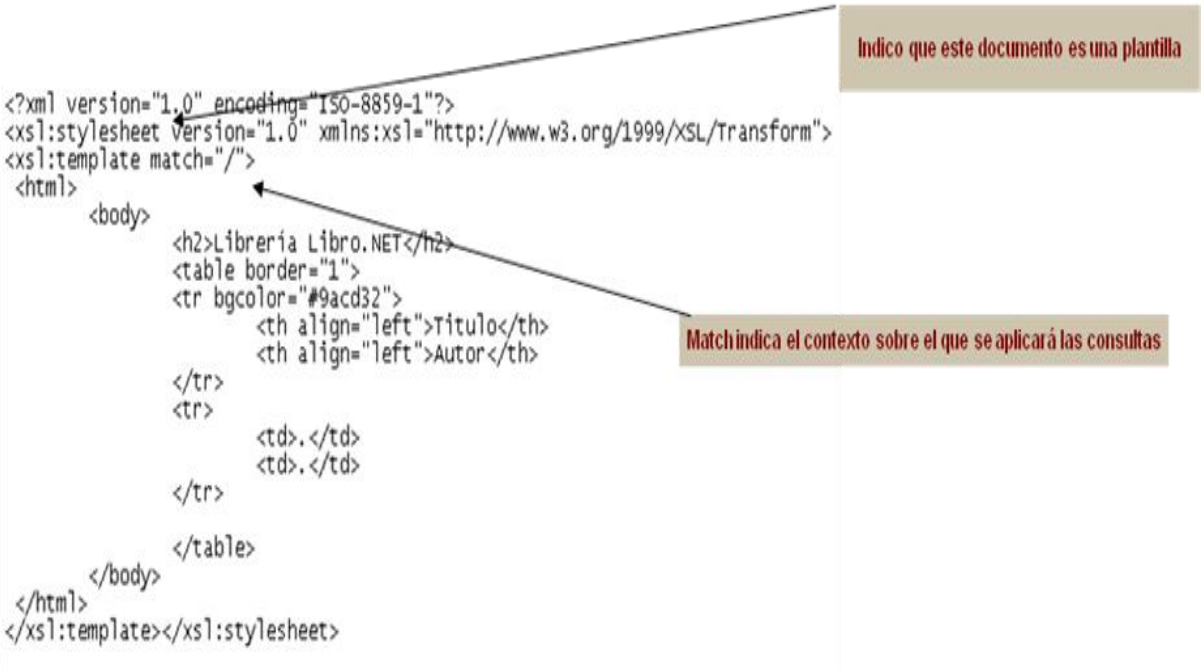
3. LAS PARTES COM EJEMPLOS

Definición de una plantilla

El elemento <xsl:template> contiene la reglas que se aplicarán cuando se encuentra un nodo coincidente. El atributo match es usado para asociar la

plantilla con un elemento XML. Este atributo puede también ser usado para definir una plantilla para una rama de un documento XML. Así, por ejemplo, match="/" define el documento entero.

Mostremos un ejemplo de declaración de una plantilla que no hace nada.



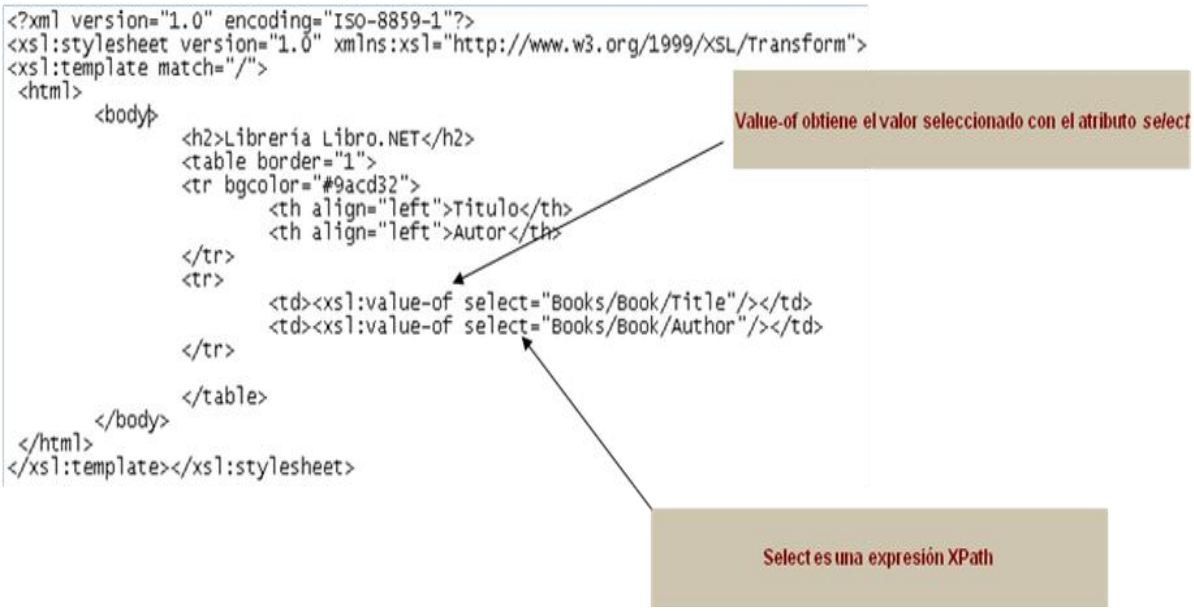

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:template match="/">
 <html>
       <body>
              <h2>Librería Libro.NET</h2>
              <table border="1">
              <tr bgcolor="#9acd32">
                     <th align="left">Titulo</th>
                     <th align="left">Autor</th>
              </tr>
              <tr>
                     <td>.</td>
                     <td>.</td>
              </tr>

              </table>
       </body>
 </html>
</xsl:template></xsl:stylesheet>
```

El elemento <xsl:value-of>

Este elemento es usado para seleccionar el valor de un elemento XML y añadirlo a la salida de la transformación. El valor del atributo **select** es una expresión XPath como las vistas anteriormente. Es decir, que obtiene los elementos seleccionados navegando en el documento fuente.

Un ejemplo de inclusión de la etiqueta value-of es mostrada a continuación. El resultado de aplicar esta hoja de estilos nos devuelve únicamente un titulo y un autor. Para poder añadir todos los títulos y autores se debe incluir un elemento que hace las funciones de bucle.



El elemento <xsl:for-each>

Este elemento es usado para seleccionar todos los elementos XML de un nodo o conjunto de nodos. Este hace las funciones de bucle que se repite para cada instancia encontrada de un nodo. El siguiente ejemplo muestra la salida de todos los títulos y autores del documento XML.

```
<?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>Librería Libro.NET</h2>
                <table border="1">
                <tr bgcolor="#9acd32">
                        <th align="left">Titulo</th>
                        <th align="left">Autor</th>
                </tr>
                <xsl:for-each select="Books/Book">
                <tr>
                        <td><xsl:value-of select="Title"/></td>
                        <td><xsl:value-of select="Author"/></td>
                </tr>
                </xsl:for-each>

                </table>
        </body>
 </html>
</xsl:template></xsl:stylesheet>
```

La salida después de aplicar esta plantilla es:

Librería Libro.NET

Titulo	Autor
Inside Microsoft SQL Server 2000	Delaney, Kalen
.NET Common Language Runtime	Burton, Kevin
C# Design Patterns	Cooper, James W.

Filtrar elementos:

Para filtrar los elementos deseados se usa la sintaxis de XPath para filtrar con la cláusula <xsl:for each>. Los operadores de filtrado que siguen el estándar son los siguientes:

- = (equal)
- != (distinto)
- < menor que
- > mayor que

El siguiente ejemplo muestra los elementos cuyo autor es *Kevin Burton*.Es ejemplo no tiene código asociado.

```
<?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>Librería Libro.NET</h2>
                <table border="1">
                <tr bgcolor="#9acd32">
                        <th align="left">Titulo</th>
                        <th align="left">Autor</th>
                </tr>
                <xsl:for-each select="Books/Book[Author='Burton, Kevin']">
                <tr>
                        <td><xsl:value-of select="Title"/></td>
                        <td><xsl:value-of select="Author"/></td>
                </tr>
                </xsl:for-each>

                </table>
        </body>
 </html>
</xsl:template></xsl:stylesheet>
```

La salida después de aplicar esta plantilla es:

Librería Libro.NET

Titulo	Autor
.NET Common Language Runtime	Burton, Kevin

Ordenación.

Se puede ordenar la salida de los elementos seleccionados por la plantilla. Esto se hace añadiendo un elemento *xsl:sort* dentro del elemento *for-each*. El siguiente ejemplo muestra la salida ordenada por un nuevo elemento que es *Publisher*. Con el atributo *select* se indica el elemento sobre el que se ordenará.

```
<?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>Libreria Libro.NET</h2>
              <table border="1">
              <tr bgcolor="#9acd32">
                     <th align="left">Titulo</th>
                     <th align="left">Autor</th>
                     <th align="left">Editor</th>
              </tr>
              <xsl:for-each select="Books/Book">
              <xsl:sort select="Publisher"/>

              <tr>
                     <td><xsl:value-of select="Title"/></td>
                     <td><xsl:value-of select="Author"/></td>
                     <td><xsl:value-of select="Publisher"/></td>
              </tr>
              </xsl:for-each>

              </table>
       </body>
 </html>
</xsl:template></xsl:stylesheet>
```

Sort ordena la salida por el elemento asignado al atributo select

Librería Libro.NET

Titulo	Autor	Editor
C# Design Patterns	Cooper, James W.	Addison Wesley
Inside Microsoft SQL Server 2000	Delaney, Kalen	Microsoft Press
.NET Common Language Runtime	Burton, Kevin	Sams

Condicionales

Se pueden usar condicionales dentro del documento para seleccionar que se quiere mostrar en la salida. Para poner una condicional simplemente hay que añadir la etiqueta <xsl:if> al documento. El valor del atributo *test* contiene la expresión que se desea evaluar.

El siguiente ejemplo muestra únicamente los elementos que tiene un número de páginas = 1088

```
<?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>Librería Libro.NET</h2>
                <table border="1">
                <tr bgcolor="#9acd32">
                        <th align="left">Titulo</th>
                        <th align="left">Autor</th>
                        <th align="left">Editor</th>
                </tr>
                <xsl:for-each select="Books/Book">
                <xsl:if test="@Pages=1088">
                <tr>
                        <td><xsl:value-of select="Title"/></td>
                        <td><xsl:value-of select="Author"/></td>
                        <td><xsl:value-of select="Publisher"/></td>
                </tr>
                </xsl:if>
                </xsl:for-each>

                </table>
        </body>
 </html>
</xsl:template></xsl:stylesheet>
```

Librería Libro.NET

Titulo	Autor	Editor
Inside Microsoft SQL Server 2000	Delaney, Kalen	Microsoft Press

Condicionales compuestos

Para poder establecer condicionales del estilo switch de C, se utiliza el elemento <xsl:choose> <xsl:when> ..<xsl:otherwise>. El siguiente ejemplo muestra su utilización.

```
<?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>Librería Libro.NET</h2>
                <table border="1">
                <tr bgcolor="#9acd32">
                        <th align="left">Titulo</th>
                        <th align="left">Autor</th>
                        <th align="left">Editor</th>
                </tr>
                <xsl:for-each select="Books/Book">
                <tr>
                        <xsl:choose>
                                <xsl:when test="Author='Delaney, Kalen'">
                                        <td bgcolor="#0000ff"> <xsl:value-of select="Title"/></td>
                                </xsl:when>
                                <xsl:when test="Author='Burton, Kevin'">
                                        <td bgcolor="#ff00ff"> <xsl:value-of select="Title"/></td>
                                </xsl:when>
                                <xsl:otherwise>
                                        <td bgcolor="#ff0000"> <xsl:value-of select="Title"/></td>
                                </xsl:otherwise>
                        </xsl:choose>

                        <td><xsl:value-of select="Author"/></td>
                        <td><xsl:value-of select="Publisher"/></td>
                </tr>

                </xsl:for-each>

                </table>
        </body>
 </html>
</xsl:template>
</xsl:stylesheet>
```

Librería Libro.NET

Titulo	Autor	Editor
Inside Microsoft SQL Server 2000	Delaney, Kalen	Microsoft Press
.NET Common Language Runtime	Burton, Kevin	Sams
C# Design Patterns	Cooper, James W.	Addison Wesley

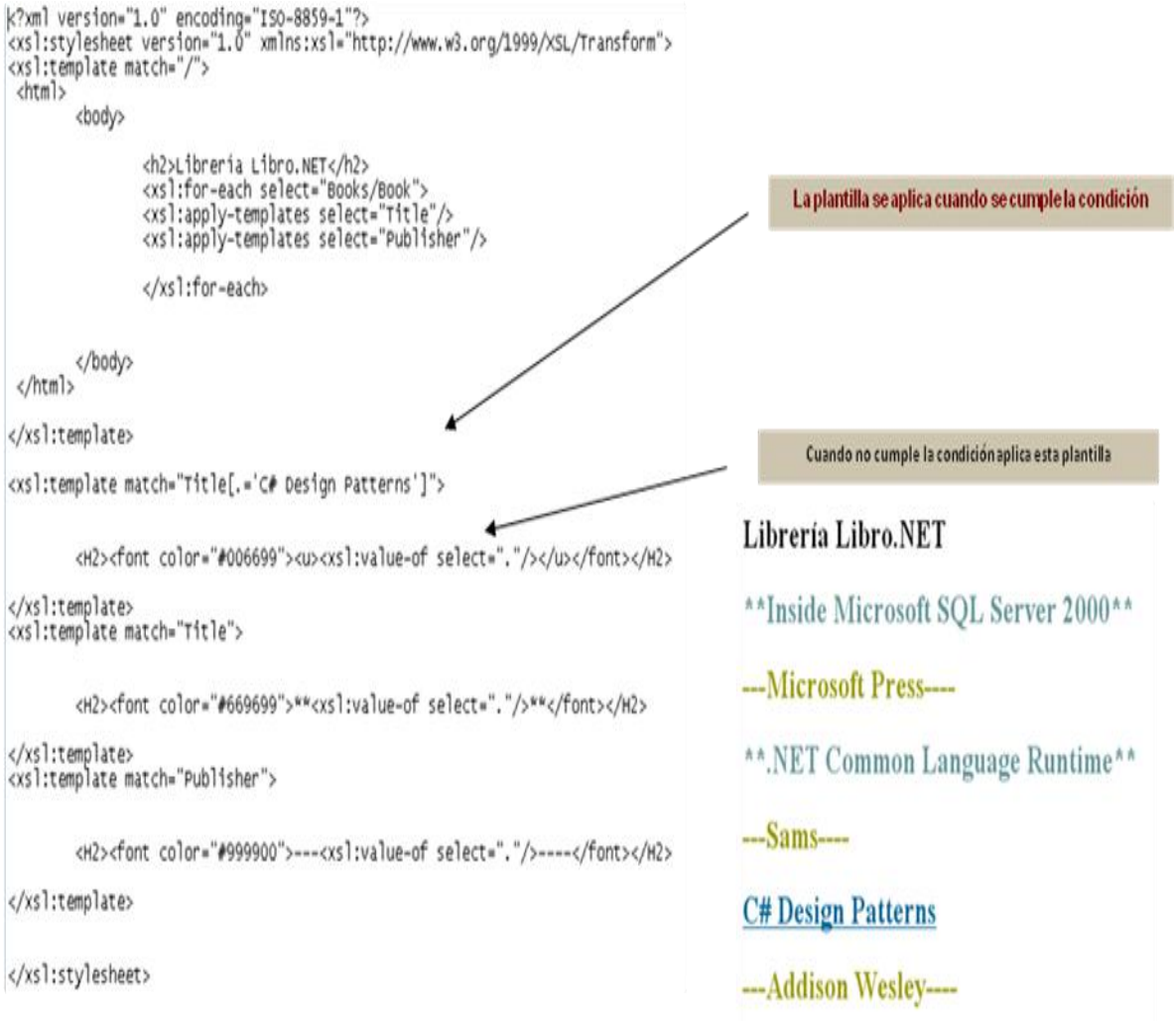
Hasta ahora todas las hojas de estilo que hemos visto incluían una única plantilla que se aplicaba al elemento raíz de documento. XSL también permite utilizar varias plantillas en una hoja de estilos. Esto es necesario por dos razones. Primero, permite ordenar la lógica de presentación del contenido del documento, haciendo más sencilla su modificación. Segundo, permite utilizar expresiones XPath para aplicar diferentes formatos a los datos XML dependiendo de su valor. Cuando una hoja de estilos contiene varias plantillas se utiliza una lógica de presentación con el comando <xsl:apply-templates>.

Normalmente se utiliza una plantilla de nivel superior para procesar el documento como un todo y se utiliza el comando apply-templates para procesar los elementos dentro del ámbito de la plantilla de nivel superior.

Si se utiliza el atributo *select* entonces se aplica la plantilla a los hijos que encajen con el valor de atributo. El atributo *select* se suele utilizar para indicar en que orden los elementos son procesados.

El siguiente ejemplo muestra su utilización. Aplica diferentes plantillas a los libros con título Title[.='C# Design Patterns']“ y a los Publisher.

Como ha podido observarse en el estudio de los ejemplo previos, generar plantillas XSLT necesita un entorno que permita fácilmente comprobar la corrección de la salida obtenida. Hay varios entorno para este propósito. Uno de ellos es <Oxygen/> (www.oxygenxml.com) que facilita la depuración de plantillas en un entorno fácil de manejar.



5. BIBLIOGRAFÍA

[XML desde .NET]: http://zarza.usal.es/~fgarcia/docencia/poo/04-05/Trabajos/XML.NET.pdf

[.NET] Framework: http://msdn.microsoft.com/en-us/magazine/cc302158.aspx

[XML tutorial]: http://www.programacion.net/html/xml/

[XML2] Leer XML con XMLTextReader en el aula. Nº 26. MARZO 2011. www.sociedadelainformacion.com

[XML3] XMLNode y XMLDocument en el aula. Nº 27. ABRIL 2011. www.sociedadelainformacion.com

[XML4] Escribiendo XMLTextWriter en el aula. Nº 27. ABRIL 2011. www.sociedadelainformacion.com

[XML5] XMLDocument en el aula . Nº 28. MAYO 2011. www.sociedadelainformacion.com

[XML6] XPath en el aula. Nº 31. AGOSTO 2011. www.sociedadelainformacion.com

SOCIEDAD DE LA INFORMACION

www.sociedadelainformacion.com

Edita:

Director: José Ángel Ruiz Felipe

Jefe de publicaciones: Antero Soria Luján

D.L.: AB 293-2001

ISSN: 1578-326x