<?xml version="1.0" encoding="utf-8"?>

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

**<xsl:template match="/módulo/profesor">**

**<patrón num="1"/>**

**</xsl:template>**

**<xsl:template match="/módulo/profesor">**

**<patrón num="2"/>**

**</xsl:template>**

</xsl:stylesheet>

La plantilla contiene un atributo match con el elemento "/módulo/profesor" del XML. Dentro de ella se crea el elemento "patrón" con el atributo "num" establecido en "1". Sin embargo, al tener otro atributo match con el mismo patron y atributo este se establecerá al ultimo indicado el cual mostrara por pantalla.

-----------------------------------------------------------------------------------------------------------------------------------------------

<?xml version="1.0" encoding="utf-8"?>

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

**<xsl:template match="/módulo/profesor" priority="1">**

**<patrón num="1"/>**

**</xsl:template>**

**<xsl:template match="/módulo/profesor">**

**<patrón num="2"/>**

**</xsl:template>**

</xsl:stylesheet>

En este caso la primera plantilla contiene un en su atributo match con un “priority” para indicar su prioridad por lo cual lo mostrará por pantalla.

<?xml version="1.0" encoding="UTF-8"?>

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

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

**<html>**

**<head></head>**

**<body>**

**<xsl:apply-templates />**

**</body>**

**</html>**

**</xsl:template>**

**<xsl:template match="módulo" >**

**<h2><xsl:value-of select="text()" /></h2>**

**</xsl:template>**

</xsl:stylesheet>

La plantilla principal crea en la raíz del XML un HTML básico con su <head> y <body>.

En el interior de este ultimo contiene una plantilla en la cual indica que se crea una etiqueta <h2> y el contenido de texto introducido con <xsl:value-of select="text()"/> sera estilo h2.

-----------------------------------------------------------------------------------------------------------------------------------

<?xml version="1.0" encoding="UTF-8"?>

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

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

**<html>**

**<head></head>**

**<body>**

**<xsl:apply-templates />**

**<ul>**

**<xsl:apply-templates />**

**</ul>**

**</body>**

**</html>**

**</xsl:template>**

**<xsl:template match="módulo" >**

**<h2><xsl:value-of select="text()" /></h2>**

**</xsl:template>**

**<xsl:template match="módulo" >**

**<li><xsl:value-of select="text()" /></li>**

**</xsl:template>**

</xsl:stylesheet>

En este caso en el interior contiene una primera plantilla en la cual indica que se crea una etiqueta <h2> y el contenido de texto introducido con <xsl:value-of select="text()"/> sera estilo h2, mientras que en la segunda indica el contenido del texto <xsl:value-of select="text()"/> sera estilo lista. Sin embargo, al llamarse iguales las plantillas la edición sera como indica la segunda.

<?xml version="1.0" encoding="UTF-8"?>

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

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

**<html>**

**<head></head>**

**<body>**

**<xsl:apply-templates mode="encabezados"/>**

**<ul>**

**<xsl:apply-templates mode="lista"/>**

**</ul>**

**</body>**

**</html>**

**</xsl:template>**

**<xsl:template match="módulo" mode="encabezados" >**

**<h2><xsl:value-of select="text()" /></h2>**

**</xsl:template>**

**<xsl:template match="módulo" mode="lista">**

**<li><xsl:value-of select="text()" /></li>**

**</xsl:template>**

</xsl:stylesheet>

En este caso se corrige el problema del modelo anterior al añadirse una etiqueta mode para diferencia cada elemento match y asi permitir controlar de manera más granular cómo se aplican las plantillas según diferentes contextos de transformación en un documento XML.

<?xml version="1.0" encoding="UTF-8"?>

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

**<xsl:template match="/módulo">**

**<xsl:apply-templates/>**

**</xsl:template>**

**<xsl:template match="profesor">**

**<patrón num="1"/>**

**</xsl:template>**

**<xsl:template match="\*">**

**<patrón num="2"/>**

**</xsl:template>**

</xsl:stylesheet>

La plantilla principal aplica la instrucción <xsl:apply-templates/>, que busca y aplica las plantillas correspondientes a los nodos hijos del elemento "módulo".

La primera plantilla crea un elemento "patrón" con el atributo "num" establecido en "1" para los elemento <profesor> del documento XML.

La segunda plantilla contiene un atributo match="\*" que indica que se aplique a todos los elementos sin plantilla asignada en el documento XML. Esta crea un elemento "patrón" con el atributo "num" establecido en "2".

<?xml version="1.0" encoding="utf-8"?>

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

**<xsl:template match="/módulo">**

**<xsl:copy/>**

**</xsl:template>**

</xsl:stylesheet>

La plantilla principal <xsl:template match="/módulo"> aplica sobre elemento raíz <módulo>. del documento XML la plantilla instrucción <xsl:copy/> que simplemente copia el elemento "módulo" por lo que devolvera en la transformacion unicamente a <modulo>.

**<?xml version="1.0" encoding="UTF-8"?>**

**<módulo/>**

-----------------------------------------------------------------------------------------------------------------------------------

<?xml version="1.0" encoding="utf-8"?>

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

**<xsl:template match="/módulo">**

**<xsl:copy-of select="."/>**

**</xsl:template>**

</xsl:stylesheet>

En este caso, la plantilla principal <xsl:template match="/módulo"> contiene la instrucción <xsl:copy-of select="."/>. Esta instrucción copia todo el contenido del elemento "módulo" por lo que esta vez si que devolverá en la transformacion a los elementos hijos de modulo.

**<?xml version="1.0" encoding="UTF-8"?>**

**<módulo>**

**<profesor>Xaime Louzán </profesor>**

**</módulo>**

<?xml version="1.0" encoding="UTF-8"?>

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

<xsl:output encoding="UTF-8" indent="yes" method="html"/>

<xsl:template match="/">

**<html>**

**<head/>**

**<body>**

**<table border="1">**

**<tr bgcolor="gray">**

**<th>Alumno</th>**

**<th>Nota</th>**

**</tr>**

**<xsl:apply-templates select="alumnos/alumno"/>**

**</table>**

**</body>**

**</html>**

</xsl:template>

<xsl:template match="alumno">

**<xsl:element name="tr">**

**<xsl:if test="nota &lt; 5">**

**<xsl:attribute name="bgcolor">red</xsl:attribute>**

**</xsl:if>**

**<td>**

**<xsl:value-of select="nome"/>**

**</td>**

**<td>**

**<xsl:value-of select="nota"/>**

**</td>**

**</xsl:element>**

</xsl:template>

</xsl:stylesheet>

La plantilla principal crea un documento HTML básico con las etiquetas <html>, <head> y <body> en la raíz del documento XML. En el <body>, se crea una <table>. La tabla contiene una fila <tr> la cual tendrá un background gris <tr bgcolor="gray">, y celdas encabezado <th> que contienen los textos "Alumno" y "Nota".

Posteriormente se indica una plantilla para los elementos <alumno> la cual indica que en el etiqueta <tr> utiliza la instrucción <xsl:if> para verificar si la nota del alumno es menor que 5. Si se cumple esa fila tendrá un background de color rojo. Finalmente, se le indica que en la primera celda contendrá el texto del elemento <nome> y la segunda celda el elemento <nota>

<?xml version="1.0" encoding="UTF-8"?>

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

<xsl:output encoding="UTF-8" indent="yes" method="html"/>

<xsl:template match="/">

**<html>**

**<head/>**

**<body>**

**<table border="1">**

**<tr bgcolor="gray">**

**<th>Alumno</th>**

**<th>Nota</th>**

**</tr>**

**<xsl:apply-templates select="alumnos/alumno"/>**

**</table>**

**</body>**

**</html>**

</xsl:template>

<xsl:template match="alumno">

**<xsl:element name="tr">**

**<xsl:choose>**

**<xsl:when test="nota &lt; 5">**

**<xsl:attribute name="bgcolor">red</xsl:attribute>**

**</xsl:when>**

**<xsl:when test="nota &lt; 7">**

**<xsl:attribute name="bgcolor">yellow</xsl:attribute>**

**</xsl:when>**

**<xsl:otherwise>**

**<xsl:attribute name="bgcolor">green</xsl:attribute>**

**</xsl:otherwise>**

**</xsl:choose>**

**<td>**

**<xsl:value-of select="nome"/>**

**</td>**

**<td>**

**<xsl:value-of select="nota"/>**

**</td>**

**</xsl:element>**

</xsl:template>

</xsl:stylesheet>

En este caso, la plantilla para los elementos <alumno> se le indica que en el etiqueta <tr> utiliza la instrucción <xsl:choose> la cual establece condiciones según la nota del alumno utilizando las instrucciones <xsl:when> y <xsl:otherwise>. Si la nota es menor a 5, un background rojo, si la nota es menor a 7, un background amarillo y en cualquier otro caso, un background verde. Finalmente, se le indica que en la primera celda contendrá el texto del elemento <nome> y la segunda celda el elemento <nota>

<?xml version="1.0" encoding="UTF-8"?>

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

<xsl:output encoding="UTF-8" indent="yes" method="html"/>

<xsl:template match="/">

**<html>**

**<head/>**

**<body>**

**<table border="1">**

**<tr bgcolor="gray">**

**<th>Alumno</th>**

**<th>Nota</th>**

**</tr>**

**<xsl:for-each select="alumnos/alumno">**

**<xsl:element name="tr">**

**<xsl:choose>**

**<xsl:when test="nota &lt; 5">**

**<xsl:attribute name="bgcolor">red</xsl:attribute>**

**</xsl:when>**

**<xsl:when test="nota &lt; 7">**

**<xsl:attribute name="bgcolor">yellow</xsl:attribute>**

**</xsl:when>**

**<xsl:otherwise>**

**<xsl:attribute name="bgcolor">green</xsl:attribute>**

**</xsl:otherwise>**

**</xsl:choose>**

**<td>**

**<xsl:value-of select="nome"/>**

**</td>**

**<td>**

**<xsl:value-of select="nota"/>**

**</td>**

**</xsl:element>**

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

**</table>**

**</body>**

**</html>**

</xsl:template>

</xsl:stylesheet>

En este caso, en vez de utilizar una plantilla para cada elemento <alumno> se utiliza un bucle <xsl:for-each select="alumnos/alumno"> para indicar que según la nota del alumno utilizando las instrucciones <xsl:when> y <xsl:otherwise> dentro del <xsl:choose>. Si la nota es menor a 5, un background rojo, si la nota es menor a 7, un background amarillo y en cualquier otro caso, un background verde. Finalmente, se le indica que en la primera celda contendrá el texto del elemento <nome> y la segunda celda el elemento <nota>.

<?xml version="1.0" encoding="UTF-8"?>

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

<xsl:output encoding="UTF-8" indent="yes" method="html"/>

<xsl:template match="/">

**<html>**

**<head/>**

**<body>**

**<table border="1">**

**<tr bgcolor="gray">**

**<th>Alumno</th>**

**<th>Nota</th>**

**</tr>**

**<xsl:for-each select="alumnos/alumno">**

**<xsl:sort select="nota" order="descending" data-type="number" />**

**<xsl:element name="tr">**

**<xsl:choose>**

**<xsl:when test="nota &lt; 5">**

**<xsl:attribute name="bgcolor">red</xsl:attribute>**

**</xsl:when>**

**<xsl:when test="nota &lt; 7">**

**<xsl:attribute name="bgcolor">yellow</xsl:attribute>**

**</xsl:when>**

**<xsl:otherwise>**

**<xsl:attribute name="bgcolor">green</xsl:attribute>**

**</xsl:otherwise>**

**</xsl:choose>**

**<td>**

**<xsl:value-of select="nome"/>**

**</td>**

**<td>**

**<xsl:value-of select="nota"/>**

**</td>**

**</xsl:element>**

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

**</table>**

**</body>**

**</html>**

</xsl:template>

</xsl:stylesheet>

En este caso, en vez de utilizar una plantilla para cada elemento <alumno> se utiliza un bucle <xsl:for-each select="alumnos/alumno"> para indicar que según la nota del alumno utilizando las instrucciones <xsl:when> y <xsl:otherwise> dentro del <xsl:choose>. Si la nota es menor a 5, un background rojo, si la nota es menor a 7, un background amarillo y en cualquier otro caso, un background verde. Ademas, Dentro del bucle, se utiliza la instrucción <xsl:sort> para ordenar los <alumno> según su nota descendentemente (menor a mayor). Finalmente, se le indica que en la primera celda contendrá el texto del elemento <nome> y la segunda celda el elemento <nota>.

<?xml version="1.0" encoding="UTF-8"?>

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

<xsl:output encoding="UTF-8" indent="yes" method="xml"/>

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

**<xsl:element name="almacenamento">**

**<xsl:element name="discos">**

**<xsl:attribute name="num">**

**<xsl:value-of select="count(//disco)"/>**

**</xsl:attribute>**

**<xsl:copy-of select="equipos/máquina/hardware/disco"/>**

**</xsl:element>**

**<xsl:element name="memorias">**

**<xsl:attribute name="num">**

**<xsl:value-of select="count(//memoria)"/>**

**</xsl:attribute>**

**<xsl:copy-of select="equipos/máquina/hardware/memoria"/>**

**</xsl:element>**

**</xsl:element>**

**</xsl:template>**

</xsl:stylesheet>

Se crea un nuevo elemento XML llamado "almacenamento" el cual contendra los elementos <discos> y <memorias>.

El elemento <disco> tendrá un atributo llamado “num” el cual su valor será el numero de discos que haya en el primer XML en la dirección "equipos/máquina/hardware/disco" los cuales contará con "count(//disco)". Posteriormente copiara los discos del primer XML y los añadirá dentro de este nuevo XML con la etiqueta <xsl:copy-of select="equipos/máquina/hardware/disco"/>.

Lo mismo para el elemento <memorias>.

<?xml version="1.0" encoding="UTF-8"?>

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

<xsl:output encoding="UTF-8" indent="yes" method="xml"/>

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

**<almacenamento>**

**<discos num="{count(equipos/máquina/hardware/disco)}">**

**<xsl:apply-templates select="equipos/máquina" mode="discos"/>**

**</discos>**

**<memorias num="{count(equipos/máquina/hardware/memoria)}">**

**<xsl:apply-templates select="equipos/máquina" mode="memorias"/>**

**</memorias>**

**</almacenamento>**

**</xsl:template>**

**<xsl:template match="máquina" mode="discos">**

**<xsl:copy-of select="hardware/disco"/>**

**</xsl:template>**

**<xsl:template match="máquina" mode="memorias">**

**<xsl:copy-of select="hardware/memoria"/>**

**</xsl:template>**

</xsl:stylesheet>

En este caso, en vez de utilizar <xsl:element name="almacenamento"> / <xsl:element name="discos"> / <xsl:element name="memorias"> para crear susodichos elementos, se crea la etiqueta directa y dentro de estas una plantilla la cual copiará los discos del primer XML y los añadirá dentro de este nuevo XML.