V Congreso de Ingeniería de Organización
Valladolid-Burgos, 4-5 Septiembre 2003

Índice

Propuesta de DTD de un lenguaje de modelado matemático basado en XML.

Marcos Calle Suárez[1], Gabriel Villa Caro[2], José Manuel García Sánchez[3]
[1] Becario F.P.I. (Ingeniero de Organización Industrial, Escuela Superior de Ingenieros – Universidad de Sevilla, Camino de los Descubrimientos, s/n. Isla de la Cartuja 41092 Sevilla, mcalle@esi.us.es)
[2]Profesor asociado (Ingeniero Industrial, Escuela Superior de Ingenieros – Universidad de Sevilla, Camino de los Descubrimientos, s/n. Isla de la Cartuja 41092 Sevilla, gvilla@esi.us.es)
[3]Profesor asociado (Ingeniero Informático, Escuela Superior de Ingenieros – Universidad de Sevilla, Camino de los Descubrimientos, s/n. Isla de la Cartuja 41092 Sevilla, jmgs@esi.us.es)

RESUMEN

Este trabajo presenta una propuesta de lenguaje para la investigación operativa basado en XML[4] e inspirado parcialmente en MathML[1], el cual se denomina ORML (Operations Research Markup Language) y se clasifica dentro de los lenguajes matemáticos compactos. Con el presente trabajo se ha estudiado la viabilidad de la creación de un lenguaje estándar de modelado, el cual posibilite la conversión desde dicho lenguaje hasta los lenguajes existentes. ORML no trata de convertirse en un sustituto de MathML[1], sino que trata de cubrir algunos aspectos no resueltos por MathML[1].

***Palabras clave**: Lenguaje, Modelado, Investigación operativa, XML.*

1. Introducción.

El modelado matemático consiste en la representación de un determinado proceso o actividad mediante los siguientes elementos: conjuntos, parámetros, variables, restricciones y funciones objetivo. El objeto de dicho modelo persigue la optimización de una o varias funciones objetivos de tipo: maximizar los beneficios, minimizar los costes, etc.

Una vez modelado el problema objeto del estudio se hace necesaria la resolución del mismo. Originalmente aparecieron numerosos paquetes de resolución de problemas matemáticos de investigación operativa. Cada uno de los citados paquetes de resolución incorporaba su propio lenguaje para permitir la lectura y resolución del citado modelo. La principal característica común a estos lenguajes consiste en la forma expandida de representar los componentes del modelo (variables, restricciones, etc). Debido a ello, cada variable se declara de modo independiente al resto. También se hace necesaria la aparición explicita de cada función objetivo y restricción. A su vez, en dicha función objetivo o restricción debe aparecer explícitamente cada una de las variables y parámetros que conforman dicha función objetivo o restricción . De todo ello se destaca la amplitud de los modelos a resolver y la arduidad de su creación.

De la anterior situación surge la necesidad de una normalización de lenguajes de modelado, surgiendo algunos lenguajes como MPS[7]. Mediante el uso de estos lenguajes de modelado se permite la exportación de modelos a los distintos paquetes de resolución, posibilitando la resolución del mismo modelo en distintos paquetes sin necesidad de crear un nuevo modelo

especifico para cada paquete de resolución. En esta etapa continua conservándose el carácter expandido de los modelos.
En una etapa posterior aparece la necesidad de agrupar las variables y restricciones. Esto conlleva una serie de mejoras, entre las cuales reseñamos las siguientes: en primer lugar aparecen los conjuntos que actuarán como índices dentro de las variables y parámetros de tipo unidimensionales y multidimensionales, otra mejora aportada consiste en la aparición de variables y parámetros de tipo unidimensionales y multidimensionales, es decir, variables y parámetros de tipo vector o matriz, la siguiente aportación consiste en vectores de restricciones que permiten definir una batería de restricciones sobre un conjunto de variables y parámetros, sustituyéndose en cada restricción los vectores o matrices de variables, o de parámetros por sus elementos correspondientes, por último hay que destacar la aparición de operaciones sobre vectores o matrices tanto de variables como de parámetros, como por ejemplo el sumatorio (Σ). De todo lo anterior surgen algunos lenguajes compactos de modelado matemático, como por ejemplo, GAMS[3], AMPL[5], MPL[8], etc. Dichos lenguajes aportan un gran avance en el modelado, simplificando las fases de creación y mantenimiento de los modelos. Una vez creado el modelo compacto se procede a la resolución previa conversión a un formato inteligible por los paquetes de resolución.

En la actualidad nos encontramos en esta última situación, caracterizada por la existencia de una diversidad de lenguajes compactos. Esta situación se caracteriza por la inexistencia de interfaces de conversión de un lenguaje compacto a otro, imposibilitando a su vez la resolución de un modelo codificado en un lenguaje compacto mediante un paquete de resolución no compatible con dicho lenguaje. Ante esta última limitación se procede mediante la creación de un nuevo modelo codificado en otro lenguaje compacto distinto, obligando de este modo a un aprendizaje adicional del nuevo lenguaje.

El presente trabajo nace para estudiar la viabilidad de la creación de un lenguaje que aglutine todas las características comunes de los lenguajes de modelado compacto existentes. Para lo cual se han realizado los siguientes estudios previos:

- Estudio de algunos de los lenguajes compactos existentes en la actualidad, concretamente GAMS[3], AMPL[5], MPL[8], LPL[6] y AIMMS[2].

- Realización de un estudio comparativo entre los lenguajes estudiados y obtención del conjunto de características comunes a los distintos lenguajes.

- Estudio de las tecnologías de comunicación basadas en XML[4], consistente en un conjunto de lenguajes de marcas orientados hacia las comunicaciones.

- Estudio de MathML[1] (Mathematical Markup Language), consistente en un estándar de lenguaje matemático de marcas enfocado a representar información general de origen matemático. Este lenguaje de marcas incorpora las recomendaciones aprobadas por el W3C (World Wide Web Consortium). MathML[1] tiene dos enfoques de representación de la información claramente diferenciados. El primero se centra en la representación gráfica de la información, mientras que el segundo de ellos se centra en la representación conceptual de la información. ORML se beneficia de la aceptación a nivel internacional de MathML[1].

2. ORML (Operations Research Markup Language)

El presente lenguaje de modelado compacto para investigación operativa ORML consta de una serie de características comunes a los lenguajes estudiados y previamente citados. La DTD de la actual propuesta se encuantra en la dirección siguiente: http://io.us.es/mcs/orml_dtd.zip. La primera característica a destacar consiste en la división de los modelos codificados en ORML en dos archivos, el primero de ellos se corresponde con un modelo paramétrico del problema, mientras que el segundo de ellos consiste en la definición de la instancia de un problema particular, es decir, en este archivo se definen los datos característicos de un problema determinado. Esta separación de modelos dos archivos permite la reusabilidad del modelo. Esta característica permite que el archivo del modelo paramétrico sea usado tantas veces como sea necesario y sin necesidad de cambios, mientras la validez de dicho modelo permanezca en vigor. El único requisito para permitir la reusabilidad consiste en la creación de un nuevo archivo de instancia de problema cada vez que se necesite resolver una nueva situación.

2.1 Archivo de modelo paramétrico del problema

Este archivo está formado por los siguientes elementos conceptuales:

- **Set (conjunto):** consisten en agrupaciones de elementos de similar naturaleza. Estos conjuntos actúan como índice para las variables, parámetros, operaciones de sumatorio, etc. Estos índices podrán ser unidimensionales o multidimensionales. Existen operaciones especificas para la definición de conjuntos basándose otros conjuntos previamente declarados, por ejemplo, unión de conjuntos, diferencia entre conjuntos, etc. Ejemplos:

  Conjunto unidimensional:

  $$Nodos\_Origen = \{\text{“Almería”}, \text{“Granada”}, \text{“Sevilla”}\} \quad (1)$$

  Conjunto multidimensional:

  $$Rutas = \{Nodos\_Origen, Nodos\_Destino\} \quad (2)$$

  Conjunto definido sobre la unión de conjuntos:

  $$Nodos = Nodos\_Origen \cup Nodos\_Destino \quad (3)$$

- **Parameter (parámetro):** consisten en valores invariables, conocidos previamente a la resolución del modelo. Existen tres tipos de parámetros: adimensional (escalar), unidimensional (vectorial) y multidimensional (matricial). Los parámetros unidimensionales y multidimensionales se definen sobre los conjuntos anteriormente citados. Ejemplos:

  Parámetro adimensional:

  $$trabajadores = 47 \quad (4)$$

  Parámetro unidimensional:

  $$demanda_{cuatrimestre} = [50 \quad 50 \quad 50] \quad (5)$$

Parámetro multidimensional:

$$coste_{producto,trimestre} = \begin{bmatrix} 4 & 5 & 9 & 10 \\ 12 & 10 & 8 & 6 \\ 13 & 11 & 3 & 5 \end{bmatrix} \quad (6)$$

- **Variable (variable):** consisten en valores variables, no conocidos previamente a la resolución del modelo. La búsqueda de estos valores es el objetivo de estos modelos. Existen tres tipos de variables: adimensional (escalar), unidimensional (vectorial) y multidimensional (matricial). Las variables unidimensionales y multidimensionales se definen sobre los conjuntos anteriormente citados. Ejemplos:

  Variable adimensional:

  $$Trabajadores \quad (7)$$

  Variable unidimensional:

  $$Producción_{cuatrimestre} \quad (8)$$

  Variable multidimensional:

  $$Produccion_{producto,trimestre} \quad (9)$$

- **Macro (macro):** consisten en la definición de una expresión algebraica que aparece reiteradas veces en el modelo. El motivo de la existencia de este elemento del modelo se debe a la simplificación del modelo que aporta su uso. Ejemplo:

  $$Macro1 = \sum_{producto} coste_{producto} \times Producción_{producto} \quad (10)$$

- **Constraint (restricción):** consisten en una ecuación algebraica que limita los posibles valores alcanzados por las variables. Existen dos tipos de restricciones en ORML: restricciones individuales y vectores de restricciones. Ejemplos:

  Restricción individual:

  $$Restriccion1: \quad \sum_{i} X_{i,i} = p \quad (11)$$

  Vector de restricciones:

  $$Restriccion_i: \quad \sum_{j} X_{i,j} = 1 \quad (12)$$

- **Objective (función objetivo):** consisten en una expresión algebraica con tendencia a maximizar o minimizar su valor. Ejemplo:

  $$\text{MIN} \quad \sum_{i,j} a_i \times d_{i,j} \times X_{i,j} \quad (13)$$

2.2 Archivo de instancia del problema

En el actual archivo se especifican los datos correspondientes a un determinado problema. Los datos a especificar consisten en los siguientes:

- Set (conjunto): elementos que forman un conjunto.
- Parameter (parámetro): valores de los parámetros.
- Variable (variable): para cada variable se permite especificar las cotas superiores e inferiores además del valor de inicial.

3. Ejemplo

En este apartado se presenta un ejemplo consistente en el modelo teórico, modelo de ORML y modelo convertido automáticamente a AMPL[5].

3.1 Modelo teórico

El ejemplo consta de dos partes, modelo paramétrico del problema e instancia del problema.

$$Min \quad \sum_{I,J} \left(cost_{I,J} \times X_{I,J} \right)$$

Sujeto a :

$$Edges_{I,J \mid I \neq J}$$

$$X_{I,J} + X_{J,I} - routes_{I,J} \geq 0$$

$$Nodes_J$$

$$\sum_I X_{I,J} - \sum_I X_{J,I} = 0$$

$$X_{I,J} \geq 0$$

Figura 1: Modelo paramétrico del problema

Conjunto $I = \{1,2,3,4\}$

$$\text{Parámetro } cost_{I,J} = \begin{bmatrix} 9999 & 1 & 1 & 9999 \\ 1 & 9999 & 1 & 1 \\ 1 & 1 & 9999 & 1 \\ 9999 & 1 & 1 & 9999 \end{bmatrix}$$

$$\text{Parámetro } routes_{I,J} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

Figura 2: Instancia del problema

3.1 Modelo de ORML

El ejemplo consta de dos partes, modelo paramétrico del problema e instancia del problema.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE model SYSTEM "orml_model.dtd">
<model name="Chinese Postman Problem">
    <sets>
        <set name="I"/>
    </sets>
    <parameters>
        <parameter name="cost">
            <index name="I"/>
            <index name="I"/>
        </parameter>
        <parameter name="routes">
            <index name="I"/>
            <index name="I"/>
        </parameter>
    </parameters>
    <variables>
        <variable name="X" type="integer">
            <index name="I"/>
            <index name="I"/>
        </variable>
    </variables>
    <constraints>
        <constraint name="Edges" relation="ge">
            <indexesdomain>
                <index name="I" alias="i"/>
                <index name="I" alias="j"/>
            </indexesdomain>
            <conditions>
                <condition relation="ne">
                    <lhs>
                        <apply>
                            <index name="I" alias="i"/>
                        </apply>
                    </lhs>
                    <rhs>
                        <apply>
                            <index name="I" alias="j"/>
                        </apply>
                    </rhs>
                </condition>
            </conditions>
            <lhs>
                <apply>
                    <plus/>
                    <apply>
                        <ci type="variable" name="X">
                            <index name="I" alias="i"/>
                            <index name="I" alias="j"/>
                        </ci>
                    </apply>
                    <apply>
                        <ci type="variable" name="X">
                            <index name="I" alias="j"/>
                            <index name="I" alias="i"/>
                        </ci>
                    </apply>
                    <apply>
                        <times/>
                        <cn>-1</cn>
                        <ci type="parameter" name="routes">
                            <index name="I" alias="i"/>
                            <index name="I" alias="j"/>
                        </ci>
                    </apply>
                </apply>
            </lhs>
            <rhs>
                <apply>
                    <cn>0</cn>
                </apply>
            </rhs>
        </constraint>
        <constraint name="Nodes" relation="eq">
            <indexesdomain>
                <index name="I" alias="j"/>
            </indexesdomain>
            <lhs>
                <apply>
                    <plus/>
                    <apply>
                        <sum/>
                        <indexesdomain>
                            <index name="I" alias="i"/>
                        </indexesdomain>
                        <apply>
                            <ci type="variable" name="X">
                                <index name="I" alias="i"/>
                                <index name="I" alias="j"/>
                            </ci>
                        </apply>
                    </apply>
                    <apply>
                        <times/>
                        <cn>-1</cn>
                        <apply>
                            <sum/>
                            <indexesdomain>
                                <index name="I" alias="i"/>
                            </indexesdomain>
                            <apply>
                                <ci type="variable" name="X">
                                    <index name="I" alias="j"/>
                                    <index name="I" alias="i"/>
                                </ci>
                            </apply>
                        </apply>
                    </apply>
                </apply>
            </lhs>
            <rhs>
                <apply>
                    <cn>0</cn>
                </apply>
            </rhs>
        </constraint>
    </constraints>
    <objectives>
        <objective name="total_cost" type="minimize">
            <apply>
                <sum/>
                <indexesdomain>
                    <index name="I" alias="i"/>
                    <index name="I" alias="j"/>
                </indexesdomain>
                <apply>
                    <times/>
                    <ci type="parameter" name="cost">
                        <index name="I" alias="i"/>
                        <index name="I" alias="j"/>
                    </ci>
                    <ci type="variable" name="X">
                        <index name="I" alias="i"/>
                        <index name="I" alias="j"/>
                    </ci>
                </apply>
            </apply>
        </objective>
    </objectives>
</model>
```

Figura 3: Modelo paramétrico del problema en ORML

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE instance SYSTEM "orml_instance.dtd">
<instance model="Chinese Postman Problem">
    <sets>
        <set name="I">
            <setcomp element="1"/>
            <setcomp element="2"/>
            <setcomp element="3"/>
            <setcomp element="4"/>
        </set>
    </sets>
    <parameters>
        <parameter name="cost">
            <paramcomp index="I" element="1">
                <paramcomp index="J" element="1" value="9999"/>
                <paramcomp index="J" element="2" value="1"/>
                <paramcomp index="J" element="3" value="1"/>
                <paramcomp index="J" element="4" value="9999"/>
            </paramcomp>
            <paramcomp index="I" element="2">
                <paramcomp index="J" element="1" value="1"/>
                <paramcomp index="J" element="2" value="9999"/>
                <paramcomp index="J" element="3" value="1"/>
                <paramcomp index="J" element="4" value="1"/>
            </paramcomp>
            <paramcomp index="I" element="3">
                <paramcomp index="J" element="1" value="1"/>
                <paramcomp index="J" element="2" value="1"/>
                <paramcomp index="J" element="3" value="9999"/>
                <paramcomp index="J" element="4" value="1"/>
            </paramcomp>
            <paramcomp index="I" element="4">
                <paramcomp index="J" element="1" value="9999"/>
                <paramcomp index="J" element="2" value="1"/>
                <paramcomp index="J" element="3" value="1"/>
                <paramcomp index="J" element="4" value="9999"/>
            </paramcomp>
        </parameter>
        <parameter name="routes">
            <paramcomp index="I" element="1">
                <paramcomp index="J" element="1" value="0"/>
                <paramcomp index="J" element="2" value="1"/>
                <paramcomp index="J" element="3" value="1"/>
                <paramcomp index="J" element="4" value="0"/>
            </paramcomp>
            <paramcomp index="I" element="2">
                <paramcomp index="J" element="1" value="1"/>
                <paramcomp index="J" element="2" value="0"/>
                <paramcomp index="J" element="3" value="1"/>
                <paramcomp index="J" element="4" value="1"/>
            </paramcomp>
            <paramcomp index="I" element="3">
                <paramcomp index="J" element="1" value="1"/>
                <paramcomp index="J" element="2" value="1"/>
                <paramcomp index="J" element="3" value="0"/>
                <paramcomp index="J" element="4" value="1"/>
            </paramcomp>
            <paramcomp index="I" element="4">
                <paramcomp index="J" element="1" value="0"/>
                <paramcomp index="J" element="2" value="1"/>
                <paramcomp index="J" element="3" value="1"/>
                <paramcomp index="J" element="4" value="0"/>
            </paramcomp>
        </parameter>
    </parameters>
</instance>
```

Figura 4: Instancia del problema en ORML

3.3 Modelo convertido automáticamente a AMPL

El siguiente modelo se consta de dos partes, modelo paramétrico del problema e instancia del problema.

```
# MODELO:   Chinese Postman Problem
set I;

param cost {I,I};

param routes {I,I};

var X {I,I} integer ;

subject to Edges {i in I, j in I :   ( i <> j )}:
     ( X[i,j] + X[j,i] + ( (-1) * routes[i,j] ) ) >= 0;

subject to Nodes {j in I}:
     ( sum {i in I} X[i,j] + ( (-1) * sum {i in I} X[j,i] ) ) = 0;

minimize total_cost:
     sum {i in I,j in I} ( cost[i,j] * X[i,j] );
```

```
# INSTANCIA DEL MODELO:   Chinese Postman Problem

set I := 1 2 3 4 ;

param cost :=
     [1,*] 1  9999   2  1      3  1      4  9999
     [2,*] 1  1      2  9999   3  1      4  1
     [3,*] 1  1      2  1      3  9999   4  1
     [4,*] 1  9999   2  1      3  1      4  9999 ;

param routes :=
     [1,*] 1  0  2  1  3  1  4  0
     [2,*] 1  1  2  0  3  1  4  1
     [3,*] 1  1  2  1  3  0  4  1
     [4,*] 1  0  2  1  3  1  4  0 ;
```

Figura 5: Modelo del problema en AMPL

4 Conclusiones y futuras ampliaciones

Este estudio ha mostrado la viabilidad de crear un lenguaje matemático compacto para investigación operativa, el cual posibilite la conversión a los lenguajes existentes.

Debido a XML, el presente lenguaje hereda el inconveniente del tamaño de los modelos generados. Para paliar este inconveniente se desarrollará en una etapa futura un interfaz gráfico de usuario para facilitar la creación y mantenimiento de modelos e instancias de problema.

Referencias

[1] Ausbrooks, R., Buswell, S., Dalmas, S., Devitt, S., Diaz, A., Hunter, R., Smith, B., Soiffer, N., Sutor, R., Watt, S., (21 Febrero de 2001) "Mathematical Markup Language (MathML)". Version 2.0. *World Wide Web Consortium*. http://www.w3.org/TR/2001/REC-MathML2-20010221

[2] Bisschop, J., Roelofs, M., (2001) "AIMMS: The Language Reference". *Paragon Decision Technology B.V.*

[3] Brooke, A., Kendrick, D., Meeraus, A., (1988) "GAMS : a user's guide". *Scientific Press*, San Francisco

[4] (6 Octubre de 2000) "Extensible Markup Language (XML)". 2nd edn. Version 1.0. *World Wide Web Consortium*. http://www.w3.org/TR/2000/REC-xml-20001006

[5] Fourer, R., Gay, D.M., Kernighan, B.W., (1993) "AMPL : a modeling language for mathematical programming". *Danvers, MA Boyd & Fraser*

[6] HÜRLIMANN T., (Agosto de 2002) "Reference Manual for the LPL Modeling Language". Version 4.43, *Departement for Informatics, Fribourg*

[7] "Passing Your Model Using Mathematical Programming System (MPS) Format", *IBM Inc.*, http://www6.software.ibm.com/sos/features/featur11.htm

[8] (28 de octubre de 2002) "MPL Manual". *Maximal Software, Inc*. http://www.maximal-usa.com/mplman/mplwtoc.html

[9] (2 Mayo de 2001) "XML Schema". Version 1.0. *World Wide Web Consortium*. http://www.w3.org/XML/Schema

[10] (16 de Noviembre de 1999) "XSL Transformations (XSLT)" Version 1.0. *World Wide Web Consortium*. http://www.w3.org/TR/xslt