A DTD for an XML-Based Mathematical Modeling Language

Marcos Calle[1], S. Lozano[1], Kate Smith[2], Terence Kwok[2] and Juan J. Domínguez[3]

[1] Dpto. Organización Industrial y Gestión de Empresas. Escuela Superior de Ingenieros. Universidad de Sevilla. Camino de los Descubrimientos s.n., 41092 Seville, Spain
{mcalle, slozano}@us.es
[2] School of Business Systems. Monash University, Clayton, Victoria 3800, Australia
{kate.smith, terence.kwok}@infotech.monash.edu.au
[3] Dpto. Lenguaje y Sistemas Informáticos. Escuela Superior de Ingeniería. Universidad de Cádiz. C/Chile, nº 1, 11003 Cádiz, Spain
juanjose.dominguez@uca.es

Abstract. This paper contains a proposal for an XML-based modeling language aimed at operations research problems. It has been called Operations Research Markup Language (ORML). There are a number of mathematical modeling languages developed by different companies, but they are proprietary and each one has its own syntax. ORML tries to bring to this field the standardization and interoperability benefits of XML.

1 Origin

This proposal of modeling language appears in an environment characterized by the existence of multitude of modeling languages for operations research problems: GAMS [3], AMPL [5], AIMMS [2], MPL [8], LPL [6], etc. In order to develop this proposal, a thorough study has been made about them. We have obtained the main characteristics of each language that make it possible to model problems. The next step on the development was to obtain a list of the common characteristics of these languages. Next, some additional characteristics have been included into the ORML proposal. This proposal of modeling language is based on two existing standards: XML and MathML.

1.1 XML

XML (eXtensible Markup Language) [4] covers the necessity of creating structured documents, making possible to exchange information between distant and independent applications. The independence character means in the first place that the mentioned applications can be implemented in multitude of languages (VB, C++, Java, etc.), and refers in second place the platform in which the mentioned application is executed (UNIX, Windows, etc.). XML is a metalanguage because it is a language used to create other languages. In addition, XML documents are legible by both humans and computers.

P.M.A. Sloot et al. (Eds.): ICCS 2003, LNCS 2660, pp. 968–977, 2003.

1.2 MathML

MathML (Mathematical Markup Language) [1] is an XML-based language. The target of this language is to express mathematical information. MathML has a double syntax. The difference of this syntax is based on the form of representing the information. The first syntax focuses on the representation format, whereas the second syntax focuses on the content of the information.

2 ORML

In this paper, we make only a summarized description of the ORML proposal, accompanied by an illustration. The complete DTDs of this proposal are available in the following URL http://io.us.es/mcs/orml_dtd.zip.

2.1 ORML Objectives

ORML has been designed to cover the following objectives:

- To express operations research problems that are not possible to express by means of MathML.
- To facilitate the standardization of a mathematical modeling languages, because at the present time each company has its own modeling language.
- To convert problems modeled with ORML to anyone of the other mentioned languages, and as far as possible to allow the inverse conversion, that is, to translate problems in any of the existing mathematical modeling languages to ORML.

2.2 ORML Structure

The structure of a problem consists, like in most of the existing mathematical modeling languages (AMPL, AIMMS, etc) of two files: problem model and problem data instance. The problem model consists of the declaration of the problem in parametric form, that is, this model contains the declarations of the objective functions, variables, restrictions, parameters, etc, without assigning numeric values. The problem data instance consists of the data of an specific case, that is, the numerical values of the parameters, bounds of the variables, initial values of the variables (seed values). The use of this structure contributes to allow for declaring the problem model only once, whereas it may be used in multiple occasions. For the later uses of this problem, the user has to update only the problem data instance, without having to modify the problem model. In our proposal, both files, problem model and problem data instance, are XML files.

2.3 Characteristics of ORML

In this section, the main elements and characteristics of ORML are described.

2.3.1 Characteristics of the Problem Model:

The problem model will contain the following sections:

- Sets
- Parameters
- Variable
- Macros
- Restrictions
- Functions objective

Set

It consists of a certain group of elements that share some important characteristic. Example of sets: "cities", "products", "warehouse", etc.

- The sets can be one-dimensional or multidimensional.
- Creation of new sets by means of operations on existing sets. These operations are union, differentiation, intersection or Cartesian product.
- The elements of the sets can be referred in the expressions.
- The main function of the sets is to act like domain index of the parameters, restrictions, variables, summation, etc.

Parameter

It is a well-known value previous to the resolution of the problem.

- They can be simple, one-dimensional or multidimensional.
- Each one-dimensional or multidimensional parameter will have an associated domain index. This parameter will have the same dimension that its index.
- The parameters can be referred in the algebraic expressions of the macros, restrictions or functions objective.

Variable

It is an unknown value previous to the resolution of the problem.

- The variables can be simple, one-dimensional or multidimensional.
- Each one-dimensional or multidimensional variable will have an associated domain index. This variable will have the same dimension that its index.
- The variables can be referred in the algebraic expressions of the macros, restrictions or functions objective.
- The variables must be one of the following basic types: continuous, binary or integer.

Macro
It consists of an algebraic expression that appears many times in the model. A macro must be declared only one time, nevertheless It can be referred in any locations where it needs to appear.

- The macros can be used in restrictions and objective functions.

Restriction
It consists of either an equality or an inequality, which it must be fulfilled in the problem.

- The restrictions can be simple, one-dimensional or multidimensional.
- Each one-dimensional or multidimensional restriction will have an associated domain index. This restriction will have the same dimension that its index.

Objective Function
It consists of an algebraic expression in which referenced variables appear. These objective functions also show whether the expression must be maximized or minimized.

Expression
- In the expressions, a particular set element can be referred, e.g.:

$$production(t) + stock(t-1) = venta(t) + stock(t) \quad (1)$$

- The use of mathematical functions and operators is allowed in the expressions. ORML contains trigonometric, hyperbolic, inverse trigonometric, inverse hyperbolic functions, functions on sets and series. It also contains comparison and arithmetic operators.

Condition
It consists of either an equality or an inequality, which must be fulfilled in the problem to allow for the inclusion of a certain algebraic expression. These conditions can be included within the summation, products and conditional element "if-then-else".

- The logic operators (AND, OR and XOR) can be used to create more complex conditions.

2.3.2 Characteristics of the Problem Data Instances:

- Definition of parameters by means of tables. This defining form is useful for multidimensional parameters.

- Definition of bounds of the variables. This definition is optional. In a problem, some of the variables can have bounds. This defining form enables to reduce the number of constraints.
- Definition of the initial values of the variables. This is useful for some resolution software, which need initial values for resolving the problems.

2.4 MathML Imported Elements

Some of the imported elements of MathML have been modified. In this section appear two lists, the first list contains the non-modified imported elements, the second list contains the modified imported elements.

Table 1. Non-modified MathML imported elements

<abs>	<and>	<apply>	<arccos>	<arccosh>
<arccot>	<arccoth>	<arccsc>	<arccsch>	<arcsec>
<arcsech>	<arcsin>	<arcsinh>	<arctan>	<arctanh>
<cartesianproduct>	<ci>	<cn>	<cos>	<cosh>
<cot>	<coth>	<csc>	<csch>	<diff>
<divide>	<else>	<exp>	<if>	<intersect>
<ln>	<log>	<matrix>	<matrixrow>	<minus>
<or>	<plus>	<power>	<quotient>	<rem>
<root>	<sec>	<sech>	<sin>	<sinh>
<tan>	<tanh>	<then>	<times>	<union>
<xor>				

Table 2. Modified MathML imported elements

<condition>	<degree>	<product>	<set>	<sum>

3 Example

This section contains a complete example. First, the problem is expressed with mathematical notation. Then, the same problem is expressed with ORML. In both cases, the information is divided in two parts. The first part includes the problem model, whereas the second part includes the problem data instance.

3.1 Problem with Mathematical Annotation

The next two sections are the problem model and problem data instance.

Problem Model

SETS (2)

$river$

$city$

PARAMETERS

$supply_{river}$

$demand_{city}$

$distcost_{river,city}$

VARIABLES

$distribute_{river,city}$

CONSTRAINTS

source :

$$\sum_{city}\left(distribute_{river,city}\right) = supply_{river} \qquad ,\forall river$$

destination :

$$\sum_{river}\left(distribute_{river,city}\right) = demand_{city} \qquad ,\forall city$$

OBJECTIVES

totalcost :

$$Min \sum_{river}\sum_{city}\left(10000000 \times distcost_{river,city} \times distribute_{river,city}\right)$$

Problem Data Instance

$$
\begin{aligned}
&\text{SETS} \\
&\quad river = \{1,2,3,4\} \\
&\quad city = \{1,2,3,4,5\} \\
&\text{PARAMETERS} \\
&\quad supply_{river} = [50 \quad 60 \quad 50 \quad 50] \\
&\quad demand_{city} = [30 \quad 20 \quad 70 \quad 30 \quad 60] \\
&\quad distcost_{river,city} = \begin{bmatrix} 16 & 16 & 13 & 22 & 17 \\ 14 & 14 & 13 & 19 & 15 \\ 19 & 19 & 20 & 23 & 999 \\ 999 & 0 & 999 & 0 & 0 \end{bmatrix}
\end{aligned}
\tag{3}
$$

3.2 Problem with ORML Annotation

This example is shown in the appendix. It consists of two XML files, for this reason the appendix contains the following two figures:

- Figure 1. Problem model.
- Figure 2. Problem data instance.

4 Summary and Discussion

In this paper, XML and MathML have been used in the development of a proposal for a new mathematical modeling language. The syntax of the elements of this language is similar to those of existing languages (GAMS, AMPL, AIMMS, LPL and MPL). ORML tries to standardize such syntax. This proposal so far consists of two DTD: the first DTD describes the structure of the problem model, the second DTD describes the structure of the problem data instance. However, DTDs are known to have limitations:

- They cannot define the datatypes.
- They do not contain namespaces that allow to import others schemas.
- They do not specify either the maximum or minimum number of times the elements can appear.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE model SYSTEM "orml_model.dtd">
<model name="metrowater">
 <sets>
  <set name="river"/>
  <set name="city"/>
  <set name="rivercity">
   <apply>
    <cartesianproduct/>
    <ci type="set" name="river"/>
    <ci type="set" name="city"/>
   </apply>
  </set>
 </sets>
 <parameters>
  <parameter name="distcost" index="rivercity"/>
  <parameter name="supply" index="river"/>
  <parameter name="demand" index="city"/>
 </parameters>
 <variables>
  <variable name="distribute" index="rivercity" type="continuous"/>
 </variables>
 <constraints>
  <constraint name="source" index="river" relation="eq">
   <lhs>
    <apply>
     <sum/>
     <indexdomain index="city"/>
     <bvar>
      <ci type="variable" name="distribute"/>
     </bvar>
     <apply>
      <ci type="variable" name="distribute"/>
     </apply>
    </apply>
   </lhs>
   <rhs>
    <apply>
     <ci type="parameter" name="supply"/>
    </apply>
   </rhs>
  </constraint>
  <constraint name="destination" index="city" relation="eq">
   <lhs>
    <apply>
     <sum/>
     <indexdomain index="river"/>
     <bvar>
      <ci type="variable" name="distribute"/>
     </bvar>
     <apply>
      <ci type="variable" name="distribute"/>
     </apply>
    </apply>
   </lhs>
   <rhs>

    <apply>
     <ci type="parameter" name="demand"/>
    </apply>
   </rhs>
  </constraint>
 </constraints>
 <objective name="totalcost" type="minimize">
  <apply>
   <sum/>
   <indexdomain index="river"/>
   <bvar>
    <ci type="variable" name="distribute"/>
   </bvar>
   <apply>
    <sum/>
    <indexdomain index="city"/>
    <bvar>
     <ci type="variable" name="distribute"/>
    </bvar>
    <apply>
     <times/>
     <cn>10000000</cn>
     <ci type="parameter" name="distcost"/>
     <ci type="variable" name="distribute"/>
    </apply>
   </apply>
  </apply>
 </objective>
</model>
```

Fig. 1. Problem model.

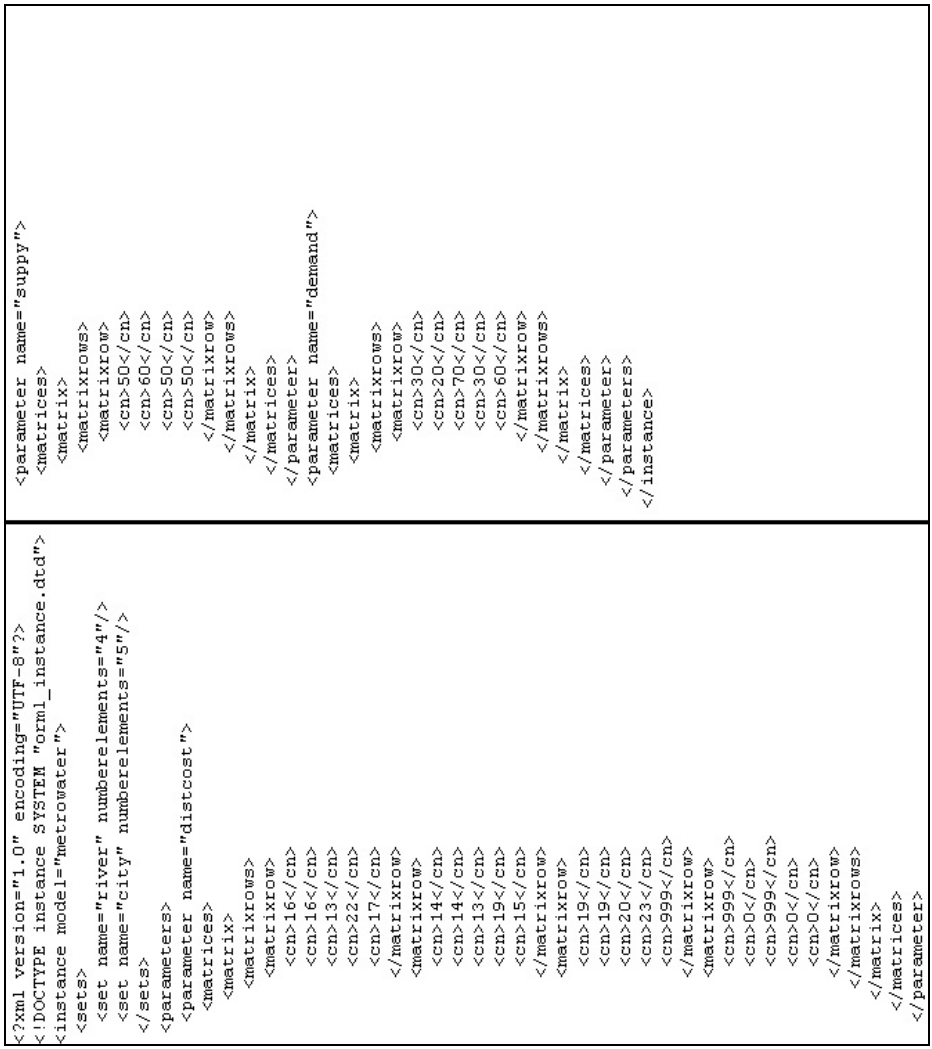

Fig. 2. Problem data instance.

Thus, the authors are working on defining appropriate XML Schemas [9]. In a later step, XSLT [10] will be used to transform ORML models to the syntax used by existing languages as well as to other formats such as MPS [7].

References

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

2. Bisschop, J., Roelofs, M.: AIMMS: The Language Reference. Paragon Decision Technology B.V. (2001)
3. Brooke, A., Kendrick, D., Meeraus, A.: GAMS: a user's guide. Scientific Press, San Francisco (1988)
4. Extensible Markup Language (XML). 2nd edn. Version 1.0. World Wide Web Consortium. http://www.w3.org/TR/2000/REC-xml-20001006 [6th October 2000]
5. Fourer, R., Gay, D.M., Kernighan, B.W.: AMPL : a modeling language for mathematical programming. Danvers, MA Boyd & Fraser (1993)
6. Hürlimann T.: Reference Manual for the LPL Modeling Language. Version 4.43, Departement for Informatics, Fribourg, [August 2002]
7. IBM Inc, "Passing Your Model Using Mathematical Programming System (MPS) Format", http://www6.software.ibm.com/sos/features/featur11.htm
8. MPL Manual. Maximal Software, Inc. http://www.maximal-usa.com/mplman/mplwtoc.html. [octubre 28th 2002]
9. XML Schema. Version 1.0. World Wide Web Consortium. http://www.w3.org/XML/Schema [2nd May 2001]
10. XSL Transformations (XSLT) Version 1.0. World Wide Web Consortium. http://www.w3.org/TR/xslt [16th November 1999]