經營科學
第17卷 第3號
2000年 11月

An XML-Based Modeling Language for the Open Trading of Decision Models

Hyoung-Do Kim*

■ Abstract ■

These days, a modeling tool or environment has to know about the others on the market and build bridges to them with which their customers insist on sharing models and data. When it is based on a closed architecture, a tangle of import/export point translators is required. Using an exchange standard, we can design an open architecture for the interchange of models and data. XML(Extensible Markup Language) provides a framework for describing the syntax for creating and exchanging data structures. The explosive growth of XML-based business proposals and standards reflects the urgent requirements and its strength. This paper proposes an XML-based language for sharing decision models within the MSOR/DSS community. The language is able to allow applications and on-line analytic processing tools to models obtained from multiple sources without having to deal with individual differences between those sources. It is expected to be a medium for B2B integration by supporting flexible interchange of decision models.

1. Introduction

These days, a modeling tool or environment has to know about the others on the market and build bridges to them with which their customers insist on sharing models and data. When it is based on a closed architecture, a tangle of import/export point translators is required. Multiple versions of translators and proprietary formats aggravate managing them in a cost-effective manner. With an exchange standard, we can solve the problem by designing an open architecture for the interchange of models and data. XML(Extensible Markup Language) provides a framework for describing the syntax for creating and exchanging data structures. The explosive growth of XML-based proposals and standards such as WIDL(Web Interface Definition Language)[1] and OTP(Open Trading Protocol)[44] reflects the urgent requirements

* Professional Graduate School of Information and Communication Technology, Ajou University

and its strength.

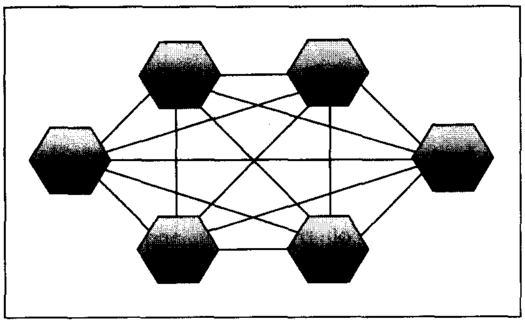
This paper proposes a structured markup language, called OOSML(Object-Oriented Structured Markup Language), for the representation and management of decision model within the MSOR/DSS community. The language is based on a conceptual modeling framework, Object-Oriented Structured Modelin(OOSM)[28], which is an extension of Structured Modeling(SM)[17]. In contrast to the previous object-oriented implementations of SM, it is an object-oriented extension to the SM framework itself, where object-oriented concepts and structuring principles such as object-oriented modular structures, models as entities and specialization are supported.

The fundamentals of OOSML are explained in this paper with a realistic and useful "initial value" of what will emerge as a comprehensive and rich collection of modeling capabilities. We expect that the language will evolve very rapidly to become a robust foundation for sharing decision models within the MSOR/DSS community. Instead of existing modeling languages such as SML[19-20], another language is required for the distinctive characteristics that XML can provide : simplicity, extensibility, interoperability, and openness. These characteristics can satisfy some new requirements for modeling tools and technologies in the age of the Internet/Web. For instance, DecisionNet[3] is such a distributed, Web-based electronic market for decision technologies such as data, models, solvers and modeling environments. Problem-specific input and output data are exchanged via HTML forms, e-mail, or the Internet file transfer protocols. However, consumers just view the results on the Web or get a results file through the Internet. By employing XML, standard developers can easily cope with dynamic changes in the process of standardization. Developers for a modeling tool or environment can implement and manage a bridge, instead of a set of bridges, in a cost-effective manner with the help of ubiquitous parsers and supporting tools. Users can share their models with others on the Internet/Web by distributing them in a standard language.

The rest of the paper is organized as follows. Section 2 discusses the value of an open architecture for the interchange of decision models. Section 3 provides an overview of the OOSML. The potential of the language is discussed in the aspect of model sharing in section 4. Finally, future research directions are summarized in section 5.

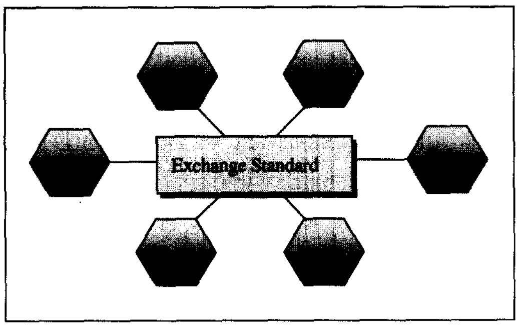
2. Open Interchange of Decision Models

In a closed architecture, a modeling tool or environment has to know about the others on the market and build bridges to them with which their customers insist on sharing models and data. This may produce a tangle of import/export point translators as <Figure 1> demonstrates. It


<Figure 1> A Web of Point Bridges

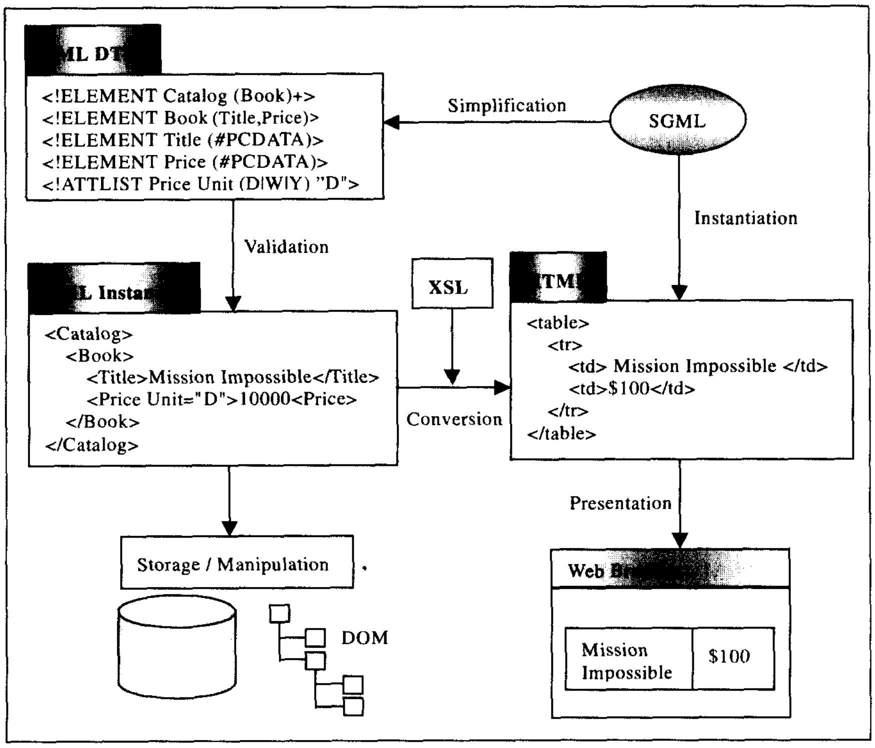
is even worse when there are many versions having different release schedules and using proprietary formats. In many cases, a bridge might not exist at all. This leaves users stranded without a way to get their models working together. Furthermore, it does not scale well.

Using an exchange standard, we can design an open architecture for the interchange of models and data, which can improve the shortcomings of the closed architecture. To participate in this architecture, each vendor only needs to add support for the standard to leverage access to all the other tools as <Figure 2> depicts. Having a standard syntax for creating and exchanging data structures is obviously important for this type of integration. XML(Extensible Markup Language) provides such a framework for describing the syntax. Furthermore, everyone can participate immediately in a Web-enabled collaborative environment.



〈Figure 2〉 Open Interchange

XML is a simplified subset of SGML[25] that maintains the SGML features of validation, structure, and extensibility. <Figure 3> demonstrates the relationship. SGML allows documents



```
<!ELEMENT Catalog (Book)+>
<!ELEMENT Book (Title,Price)>
<!ELEMENT Title (#PCDATA)>
<!ELEMENT Price (#PCDATA)>
<!ATTLIST Price Unit (D|W|Y) "D">
```

```
<Catalog>
  <Book>
    <Title>Mission Impossible</Title>
    <Price Unit="D">10000<Price>
  </Book>
</Catalog>
```

```
<table>
  <tr>
    <td> Mission Impossible </td>
    <td>$100</td>
  </tr>
</table>
```

〈Figure 3〉 XML Application Process

to be self-describing, through the specification of tag sets and the structural relationships between the tags. This specification is referred to as the Document Type Definition(DTD). HTML is a small hard-wired set of about 70 tags and 50 attributes, which allow HTML users to skip the self-describing aspect from a document. XML, on the other hand, retains the key SGML advantage of self-description through DTDs, while avoiding the complexity of full-blown SGML. XML is making rapid progress through standardization process. It has many benefits for folks who want to improve structure, maintainability, searchability, presentation, and other aspects of their document management. In addition to modifying the syntax and semantics of document tag annotations, XML also changes our linking model by allowing authors to specify different types of document relationships. Furthermore, there is a presentation specification language for XML documents that keep structuring and presentation information separate from actual data. The language XSL(Extensible Style Language) enables developers to format information more easily for Web viewing. Refer to references[6, 14, 27, 46, 50] for XML details.

Many communities have struggled to codify the tacit knowledge of their data using XML. The explosive growth of XML-based proposals and standards, inclusive of RDF(Resource Description Format)[33], SMIL(Synchronized Multimedia Integration Language)[48], CML(Chemical Markup Language)[42], MathML(Mathematical Markup Language)[24], OFX[9], CDF[12], OSD[22], OBI[43], and OTP[44], reflects the urgent requirements. The goal of this paper is also to propose such a language for sharing models within the MSOR/DSS community. The language is able to allow applications and on-line analytic processing tools to models obtained from multiple sources without having to deal with individual differences between those sources. In addition, it enables combined, collaborative use of a potentially very large number of individual models and proactive administration of collections of models based on business needs as well as mathematical principles. These capabilities are fundamental to effective deployment of decision models in commercial application domains. In the aspect of model sharing, the language is very similar to PMML (Predictive Model Markup Language). PMML provides a quick and easy way for companies to define predictive models and share models between compliant vendors' applications. A PMML document provides a non-procedural definition of fully trained or parameterized analytic models with sufficient information for an application to deploy them. By parsing it with any standard XML parser, the application can determine the types of data input to and output from the models, the detailed forms of the models, and how to interpret their results.

Someone may ask why we need another language instead of existing ones such as SML[20, 21], GAMS[7], and AMPL[15]. It can be explained by the distinctive characteristics of XML : simplicity, extensibility, interoperability, and openness. XML's rigid set of rules helps make documents more readable to both humans and machines. XML documents are built upon a core set of basic nested structures. While the structures themselves can grow complex as layers of detail are added, the mechanisms underlying those structures require very little implementation effort. XML is extensible in two senses. First, it allows developers to create their own DTDs, effectively creating 'extensible' tag sets that can be used for multiple applications. Basically, XML is a meta-language that can define a new tag-based language using XML DTD. XML-based standards and proposals are such things that are defined using XML DTDs. Second, XML itself is being extended with several additional standards that add styles, linking, and referencing ability to the core XML set of capabilities. As a core standard, XML provides a solid foundation around which other standards may grow. Interoperability means that XML can be used on a wide variety of platforms and interpreted with a wide variety of tools. Because the document structures behave consistently, parsers that interpret them can be built at relatively low cost in any of a number of languages. XML supports a number of key standards for character encoding, allowing it to be used all over the world in a number of different computing environments. Openness means that the standard itself is completely open, freely available on the Web and that anyone can parse a well-formed XML document, and validate it if a DTD is provided.

3. OOSML : A Structured Markup Language for Sharing Decision Models

In the fields of MS/OR and Decision Support Systems(DSS), modeling processes are knowledge-intensive and time-consuming. Researches on Modeling Environments(ME)[18, 45] and Model/Modelbase Management Systems(MMS) [2, 5, 23, 26, 32, 41] are active in order to support the modeling processes and related activities. To implement such an ME or MMS, a conceptual modeling framework is required for representing and managing decision models. Some distinctive frameworks for the purpose are as follows : Structured Modeling(SM)[17], logic-based modeling[2, 31], graph grammar[26], object-oriented modeling[23, 41] and frame-based modeling[4, 35, 36]. They raise modeling to a higher plane of abstraction and generality compared with traditional solvers.

OOSML is based on a conceptual modeling framework, Object-Oriented Structured Modeling(OOSM), which is an extension of SM[17]. Most of object-oriented approaches to SM are in the level of implementation. For example, Lenard[38] uses the inheritance hierarchy of object-oriented programming paradigm to represent a structured model in terms of objects and classes. In implementing an SM language called BLOOMS, Gagliardi and Spera[16] extended the definition of the 10th SM core concept (generic structure) in the framework of object orientation. Added definition is the following : Genera have their own functionality: Genera of the same type have the same operational behavior. A graphical approach taken by Chari and Sen[8] focus on a model graph for representing a model class. Note that model graphs can have module nodes that encapsulate a subgraph containing a group of related nodes. Their implementation named GBMS/SM supports all the stages of the model development life-cycle, which is implemented by an object-oriented language. Another graphical approach by Hamacher et al.[23] adds some features from the entity-relationship model to the genus graph. In contrast to the previous object-oriented implementations of SM, OOSM is an object-oriented extension to the structured modeling framework itself, where object-oriented concepts are systematically supported by structuring principles such as object-oriented modular structures, models as entities and generalization/specialization. Please refer to Kim [32] for the details of OOSM.

Based on the framework, this paper proposes a structured markup language, OOSML. The language is formally defined by a simple XML DTD which is specified in <Figure 4>. An OOSML specification is composed of one model element and any number of instance elements. A model element should be composed of one or more elements of 'model', 'entity', 'aGenus', 'vaGenus', 'fGenus', or 'tGenus'. Such a model element has an 'id' attribute for uniquely identifying itself in a document. An 'entity' element should contain either 'peGenus' or 'ceGenus'. 'aGenus', 'vaGenus', 'fGenus', or 'tGenus' elements can be added to such an entity. A 'peGenus' can be empty. That is, it can be used without an end tag, e.g., <peGenus id= PLANT/>. A 'ceGenus' element can contain 'calls' elements, whose 'genus' attribute identifies called genera. 'aGenus' and 'vaGenus' elements should contain a 'datatype' element, of which 'dt' attribute identifies one of predefined data types. A 'fGenus' element should contain one or more 'calls' elements and a 'frule' element. The 'type' attribute of a 'frule' element identifies one of predefined functional rules. Similarly, a 'tGenus' element should contain a 'trule' element.

Let's take the the Hitchcock-Koopmans transportation problem[10, 17, 38] as an illustrative example. The problem can be fully specified by OOSML as in <Figure 5>. First of all, a 'model' specification starts with a model element, which

```
<!ENTITY % nodeAttrs 'id ID #IMPLIED'>
<!ENTITY % elementAttrs 'occurs (REQUIRED|OPTIO-
              NAL|ONEORMORE|ZEROORMORE)
              "REQUIRED"'>

<!ELEMENT oosml (model,instance*)>

<!ELEMENT model (model|entity|aGenus|vaGenus|fGenus|
              tGenus)+>
<!ATTLIST model %nodeAttrs; >

<!ELEMENT entity ((peGenus|ceGenus),(aGenus|vaGenus|
              fGenus|tGenus)*)>
<!ATTLIST entity %nodeAttrs;
                        %elementAttrs; >

<!ELEMENT peGenus EMPTY>
<!ATTLIST peGenus %nodeAttrs; >

<!ELEMENT ceGenus (calls)*>
<!ATTLIST ceGenus %nodeAttrs; >

<!ELEMENT calls EMPTY>
<!ATTLIST calls genus IDREF #REQUIRED
                    %elementAttrs; >

<!ELEMENT aGenus (datatype)>
<!ATTLIST aGenus %nodeAttrs; >

<!ELEMENT vaGenus (datatype)>
<!ATTLIST vaGenus %nodeAttrs; >

<!ELEMENT datatype EMPTY>
<!ATTLIST datatype dt (R|RP|I|IP|STRING) "RP">

<!ELEMENT fGenus ((calls)+,frule)>
<!ATTLIST fGenus %nodeAttrs; >

<!ELEMENT frule EMPTY>
<!ATTLIST frule
    type (SCALE|SHIFT|POWER|MIN|MAX|LN|EXP
        |VSUM|VPROD|SUM|PROD|DOT) "SUM">

<!ELEMENT tGenus ((calls)+,trule)>
<!ATTLIST tGenus %nodeAttrs; >

<!ELEMENT trule EMPTY>
<!ATTLIST trule type (LE|LT|EQ|GT|GE) "LE">

<!ELEMENT instance ANY>
```

<Figure 4> XML DTD for OOSML

```
<?xml version="1.0"?>
<!DOCTYPE oosml SYSTEM "oosml.dtd" >
<oosml>
  <model id="M_TRANSPORTATION">
      <aGenus id="Name">
           <datatype dt="STRING" />
      </aGenus>
      <aGenus id="Creator">
           <datatype dt="STRING" />
      </aGenus>
      <entity id="E_PLANT" occurs="ONEORMORE">
           <peGenus id="PLANT" />
           <aGenus id="SUPPLY">
                <datatype dt="RP" />
           </aGenus>
           <fGenus id="OUTFLOW">
                <calls geus="FLOW"
                        occurs="ONEORMORE" />
                <frule type="SUM" />
           </fGenus>
           <tGenus id="T_SUPPLY">
                <calls genus="OUTFLOW"
                        occurs="REQUIRED" />
                <calls genus="SUPPLY"
                        occurs="REQUIRED" />
                <trule type="LE" />
           </tGenus>
      </entity>
      <entity id="E_CUSTOMER"
                occurs="ONEORMORE">
           <peGenus id="CUSTOMER" />
           <aGenus id="DEMAND">
                <datatype dt="RP" />
           </aGenus>
           <fGenus id="INFLOW">
                <calls genus="FLOW"
                  occurs="ONEORMORE" />
                <frule type="SUM" />
           </fGenus>
           <tGenus id="T_DEMAND">
                <calls genus="INFLOW"
                  occurs="REQUIRED" />
                <calls genus="DEMAND"
                  occurs="REQUIRED" />
                <trule type="EQ" />
           </tGenus>
      </entity>
      <entity id="T_LINK">
           <ceGenus id="LINK">
                <calls genus="PLANT"
                  occurs="REQUIRED" />
                <calls genus="CUSTOMER"
                  occurs="REQUIRED" />
           </ceGenus>
           <aGenus id="UNITCOST">
                <datatype dt="RP" />
           </aGenus>
           <vaGenus id="FLOW">
                <datatype dt="RP" />
           </vaGenus>
      </entity>
      <fGenus id="TOTALCOST">
           <calls genus="FLOW"
                occurs="ONEORMORE" />
           <calls genus="UNITCOST"
                occurs="ONEORMORE" />
           <frule type="DOT" />
      </fGenus>
  </model>
  <instance>…</instance>
</oosml>
```

<Figure 5> OOSML Representation of the Transportation Problem

is the target model to be described. For example, the transportation model is described by the following markup.

<model id = "M_TRANSPORTATION">
…
</model>

The 'id' atribute serves a dual role of identifying the definition, and also naming the specific model class.

A model is an aggregate of other models and entities. The model 'M_TRANSPORTATION' is an aggregate of 'E_PLANT', 'E_CUSTOMER' and 'E_LINK'. Furthermore, such a model can be described by attribute genus(aGenus), variable attribute genus(vaGenus), function genus(fGenus) and test genus(tGenus) like any entities. 'M_TRANSPORTATION' has 'Name' and 'Creator' attribute genera.

Entities group aGenus, vaGenus, fGenus and tGenus around peGenus or ceGenus, so they have to contain only one entity genus. The entity 'E_PLANT' contains four genera, one of which is a peGenus 'PLANT'. An entity may be required or optional for a model, and may occur multiple times, as indicated by its 'occurs' attribute having one of the four values 'REQUIRED', 'OPTIONAL', 'ZEROORMORE' or 'ONEORMORE'. The default value is 'REQUIRED'. One or more elements of the entity 'E_PLANT' should occur in the instances of the model 'M_TRANSPORTATION'. Attribute and variable attribute genera must have a data type of 'R', 'RP', 'I' or 'IP', where 'R' stands for real values, 'RP' for positive real values, 'I' for integer values, and 'IP' for positive integer values. Attribute genus 'SUPPLY' must have a positive real value.

Function and test genera can have 'calls' elements which explicitly define mathematical dependency on other genera. The 'genus' attribute of a 'calls' element identifies the genus called by the element. The 'calls' elements can also have an 'occurs' attribute. Related with the mathematical dependencies, 'fGenus' and 'tGenus' must have a rule for computation or comparison, respectively. The rule of a function genus has a 'type' attribute whose value is among 'SCALE', 'SHIFT', 'POWER', 'MIN', 'MAX', 'LN', 'EXP', 'VSUM', 'VPROD', 'SUM', 'DOT' or 'PROD'. Its default value is 'SUM'. The function genus 'OUTFLOW' has a rule type 'SUM' that means summation on the 'FLOW' genus. The rule of a test genus has a type attribute whose value is among 'LE', 'LT', 'GE', 'GT' or 'EQ'. Its default value is 'LE'. The function genus 'T_SUPPLY' has a rule type 'LE' that means 'less than or equal to'. In most systems and environments based on SM, mathematical knowledge is only embedded in strings, and may be interpreted or compiled for function evaluation. The specification of function genera by factorable functions[37] is an early effort to conceptualize the generic rules in DSS area. Basic principle applied to the language design is to symbolically define the generic rules of function and test genera, and then to infer or constrain other facts based on the mathematical knowledge.

In an OOSML specification, a model instance is defined by an 'instance' element. The element can contain any other elements. However, they have to follow the schema defined in the 'model' element. For example, <Figure 6> shows the way to define instance elements. All the tag

```
<instance>
        <Name>KoreaDistributionModel</Name>
        <Creator>H.D. Kim</Creator>
        <E_PLANT>
                <PLANT>DALLAS</PLANT>
                <SUPPLY>20,000</SUPPLY>
        </E_PLANT>
        <E_PLANT>
                <PLANT>CHICAGO</PLANT>
                <SUPPLY>42,000</SUPPLY>
        </E_PLANT>
        <E_CUSTOMER>
                <CUSTOMRER>NEWYORK</CUSTOMRER>
                <DEMAND>25,000</DEMAND>
        </E_CUSTOMER>
                <E_CUSTOMER>
                <CUSTOMRER>ATLANTA</CUSTOMRER>
                <DEMAND>15,000</DEMAND>
        </E_CUSTOMER>
        <E_CUSTOMER>
                <CUSTOMRER>LA</CUSTOMRER>
                <DEMAND>22,000</DEMAND>
        </E_CUSTOMER>
        <T_LINK>
                <LINK><PLANT>DALLAS</PLANT>
                <CUSTOMER>NEWYORK</CUSTOMER>
                </LINK>
                <UNITCOST>23</UNITCOST>
        </T_LINK>
        <T_LINK>
                <LINK><PLANT>DALLAS</PLANT>
                <CUSTOMER>ATLANTA</CUSTOMER>
                </LINK>
                <UNITCOST>17</UNITCOST>
        </T_LINK>
        <T_LINK>
                <LINK><PLANT>DALLAS</PLANT>
                <CUSTOMER>LA</CUSTOMER></LINK>
                <UNITCOST>32</UNITCOST>
        </T_LINK>
        <T_LINK>
                <LINK><PLANT>CHICAGO</PLANT>
                <CUSTOMER>NEWYORK</CUSTOMER>
                </LINK>
                <UNITCOST>7</UNITCOST>
        </T_LINK>
        <T_LINK>
                <LINK><PLANT>CHICAGO</PLANT>
                <CUSTOMER>ATLANTA</CUSTOMER>
                </LINK>
                <UNITCOST>23</UNITCOST>
        </T_LINK>
        < T _ L I N K >
                <LINK><PLANT>CHICAGO</PLANT>
                <CUSTOMER>LA</CUSTOMER></LINK>
                <UNITCOST>30</UNITCOST>
        </T_LINK>
</instance>
```

<Figure 6> A Model Instance Definition of the Transportation Problem

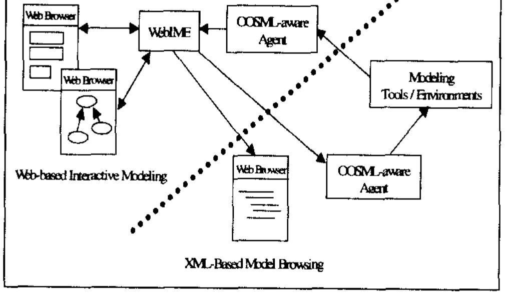
names come from the identifiers of their corresponding schema elements. Agents can validate OOSML model instances using its schema defined in an OOSML specification.

In the past years, two kinds of modeling languages have been intensively studied by researchers: (1) algebraic languages such as AMPL [15] and GAMS [7] and (2) SM languages. What makes the algebraic languages so popular in MS/OR community stems almost from the algebraic notation. Some comments and limitations on those languages can be found in [16, 39]. These motivated the study of the SM framework and variants of SM languages. OOSML can be viewed as a variant of SM, so it has the same benefits with SM languages. Those advantages include wide range of model representation and hierarchical structuring of models. In addition, OOSML adds object-oriented structuring principles into the SM framework. Those principles allow users to classify model components into entities and models that naturally match with users' view. OOSML is fundamentally different from the other SM languages in that it is based on generalized markups. At least, OOSML inherits the advantages of XML inclusive of the following; (1) It is possible to view or process OOSML models on the Web. That is, we can transfer the semantics of models into the ubiquitous browsers of users. (2) We can share OOSML DTD and style sheets through global repositories as in the area of XML/EDI [51]. (3) There are many open tools and standards such as parsers and DOM[47] API for processing OOSML models. So, we can easily build a translator between heterogeneous models, e.g., between GAMS and OOSML models.

4. Model Sharing

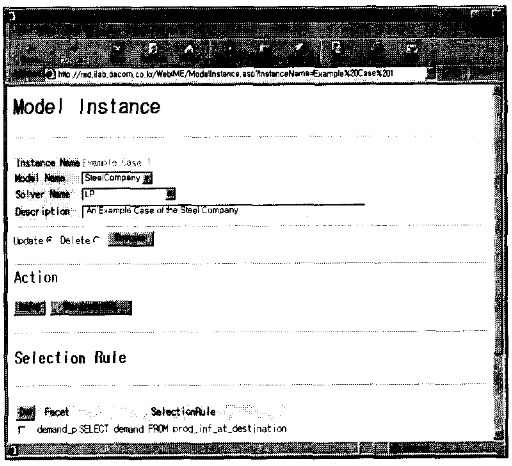
This paper has adopted XML as a meta-language for specifying OOSML, where its characteristics are described using elements and attributes. Although HTML provides a universal way to present information, it only addresses the presentation of data. XML takes this one step further by addressing the context, or meaning of the data. By defining the structure of decision models with XML tags, finding, manipulating, acting on and interacting with the models are much easier. When a user's agent gets a model that conforms to the syntax and semantics of OOSML, it is able to analyze or manipulate the model easily and correctly. The highly structured delivery of data enables open interchange between servers and clients, and potentially between servers themselves.

In the aspect of model sharing, first of all, users can manipulate OOSML models, downloaded from a Web server, using a Web browser. They can analyze such a model by applying XSL style sheets to it. <Figure 7> demonstrates a simple architecture for sharing OOSML models between a modeling environment and other modeling tools/environments. The former



<Figure 7> A Simple Architecture for OOSML-Based Model Sharing

is a Web-based integrated modeling environment(WebIME)[29] for sharing modeling knowledge on the Web. It is based on a multi-facetted modeling approach to mathematical model representation and management[30]. It includes various syntax-directed editors and graphical tools to support conceptual modeling, mathematical modeling, storage and retrieval, and solution. In the environment, we can generate OOSML models from its model base on the Web. <Figure 8> shows the model instance definition page, where OOSML models can be generated by clicking on the 'Generate XML' button. <Figure 9> demonstrates an applet viewer that uses a Java XML parser to display a generated XML document from the transportation model in a tree view.


http://rad.ilab.dacom.co.kr/WebIME/ModelInstance.asp?InstanceName=Example%20Case%201

Model Instance

Instance Name Example Case 1
Model Name SteelCompany
Solver Name LP
Description An Example Case of the Steel Company

Update ◉ Delete ○

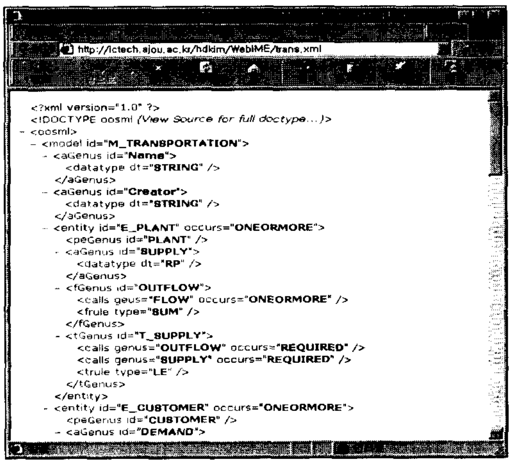
Action

Selection Rule

Facet SelectionRule
demand_p SELECT demand FROM prod_inf_at_destination

〈Figure 8〉 OOSML Model Generation in WebIME

In order to integrate modeling tools and environments using OOSML, we need to implement OOSML-aware agents. These agents are required to have four basic components : model access on the Internet/Web, OOSML model processing, interfacing with proprietary systems, and model translation. The second one is to manipulate OOSML models using XML


http://lctech.ajou.ac.kr/hdkim/WebIME/trans.xml

```
  <?xml version="1.0" ?>
  <!DOCTYPE oosml (View Source for full doctype...)>
- <oosml>
  - <model id="M_TRANSPORTATION">
    - <aGenus id="Name">
        <datatype dt="STRING" />
      </aGenus>
    - <aGenus id="Creator">
        <datatype dt="STRING" />
      </aGenus>
    - <entity id="E_PLANT" occurs="ONEORMORE">
        <peGenus id="PLANT" />
      - <aGenus id="SUPPLY">
          <datatype dt="RP" />
        </aGenus>
      - <fGenus id="OUTFLOW">
          <calls geus="FLOW" occurs="ONEORMORE" />
          <frule type="SUM" />
        </fGenus>
      - <tGenus id="T_SUPPLY">
          <calls genus="OUTFLOW" occurs="REQUIRED" />
          <calls genus="SUPPLY" occurs="REQUIRED" />
          <trule type="LE" />
        </tGenus>
      </entity>
    - <entity id="E_CUSTOMER" occurs="ONEORMORE">
        <peGenus id="CUSTOMER" />
      - <aGenus id="DEMAND">
```

〈Figure 9〉 Transportation Model in a Tree View

technologies including XML, XSL, and DOM. On the other hand, the third one is to deliver models to a specific modeling tool or environment. From parsed OOSML models, we can generate proprietary models through various ways using scripts and programs. Considering flexible management, XSL-based transformation is a good choice. <Figure 10> shows a part of XSL-based rules for transforming an OOSML model into a GAMS model. Although the sample rule looks very complex, it just identifies a peGenus contained in an entity and then finds all data related with the peGenus. Please refer to [49] for elemental details of XSL. Note that WebIME also implements an agent of the same kind to obtain OOSML models from other modeling tools and environments.

OOSML can be used for sharing decision models on the Internet among heterogeneous modeling tools and environments. Working with an open standardized interchange format makes it possible for modelers to use modeling tools appropriate for their objectives. Assuming that each tool supports an interchange format, we can trade decision models instead of exchanging

```
<?xml version="1.0"?>
<xsl:stylesheet
xmlns:xsl="http://www.w3.org/TR/WD-xsl">
        <xsl:template match="/">
  <xsl:for-each select="/oosml/model/entity">
        <xsl:if test="./peGenus">
              SET <xsl:value-of select="@id">
                  </xsl:value-of> /
              <xsl:for-each select="/oosml
               /instance/*[.!nodeName()
               =context(-1)/@id]">
                    <xsl:value-of select=
                    "*[.!nodeName()=context(-2)/
                    peGenus/@id]">

                    </xsl:value-of>
              </xsl:for-each>
              / ;
        </xsl:if>
  </xsl:for-each>
  ...
</xsl:template>
</xsl:stylesheet>
```

〈Figure 10〉 XSL-based Transformation Rule

proprietary models. OOSML is the first trial to propose an open and extensible model exchange format for modeling tools and environments.

In the DecisionNet, a software agent leads a user through a session in which(s) he supplies requested data through a series of HTML forms. All the model manipulation and output generation are performed on the server side. This approach limits the role of clients or other servers. One reason for the limitation is related with the automatic generation of model instances from corporate systems and databases. Because a user agent can't interpret the meaning of the HTML forms, especially without model schema definition, the existing systems on the Web only support model-specific solutions or require that users should have high-level knowledge about modeling language specifics. In the DecisionNet, for example, users have to develop AMPL files. On the other hand, an XML application, WIDL(Web Interface Definition Language)[1], enables automation of all interactions with HTML/XML documents and forms, providing a general method of representing request/response interactions over standard web protocols, and allowing the Web to be utilized as a universal integration platform. It enables interfaces to be described for web sites that are not controlled by calling programs. Instead of interface description, OOSML is used for specifying decision models, whose structure and semantics are contained in its DTD.

One of the desirable features of a new generation of modeling systems is the independence of model representation and model solution, with model interface standards to facilitate building a library of models and easily accessed solvers for retrieval, systems of simultaneous equations, optimization, and other important manipulations[17]. Logically, two servers for modeling and solving problems may be separate, but existing tools and environments are tightly coupled with specific solvers. OOSML is a standard language for promoting the open trading of models between servers.

5. Concluding Remarks

This paper proposes a structured markup language for model representation and management on the Web as an XML application. The language is based on a conceptual modeling framework, OOSM, which is an object-oriented extension of SM. The language supports object-oriented concepts such as object-oriented modular structure, models as entities and specialization using structured markups. Such a language will be a catalyst in trading or integrating models or model-related activities on the Web and transform the Web from a global information space into a universal knowledge network. That is, it is expected to be a medium for B2B integration by supporting flexible interchange of decision models among business organizations. We demonstrate a simple architecture for OOSML-based model sharing with some implementation details between WebIME and other modeling tools/environments. Compared with making a non-XML standard such as SML, OOSML retains the distinctive characteristics that XML can provide : simplicity, extensibility, interoperability, and openness. Compared with modeling tools/environments based on a closed architecture, ones based on OOSML only needs to add support for the standard to leverage access to all the other tools.

Further research directions include the following :

1) to standardize such a modeling language through practical reviews and modifications in the MSOR/DSS community; Standardization can remove the need to invent another XML-based modeling language that will invoke excessive overhead for translation, learning, etc. OOSML should evolve into a standard language for decision model representation and management. The standard language will be a catalyst in trading or integrating models or model-related activities on the Internet/Web.
2) to support XML-based modeling on the Web: XML technologies can be applied to modeling work itself to create an OOSML model schema and its instances. Using a model schema represented by the OOSML, for example, a user agent can perform diverse model management activities including form generation for modeling instances interactively on the Web. The highly structured delivery of data enables the agent to present different views of the same information in a cost-effective manner.

References

[1] Allen, C., WIDL: Application Integration with WIDL, http://www.webmethods.com/technology/widl.html(1997).

[2] Bhagava, H.K. and S.O. Kimbrough, On Embedded Languages for Model Management, Decision Support Systems 10, No.3 (1993), pp.277-299.

[3] Bhargava, H.K, R. Krishnan and S. Roehrig, Model Management in Electronic Markets for Decision Technologies : A Software Agent Approach, Proceedings of HICSS 30(1997).

[4] Binbasioglu, M. and M. Jarke, Domain Specific DSS Tools for Knowledge-based Model Building, Decision Support Systems 2, No.3 (1986), pp.213-223.

[5] Blanning, R.W., A Relational Framework for Join Implementation in Model Management Systems, Decision Support Systems 1, No.1(1985), pp.69-82.

[6] Bray, T., J. Paoli and C.M. Sperberg-McQueen, Extensible Markup Language (XML) 1.0, http://www.w3.org/TR/1998/REC-xml-19980210(1998)

[7] Brooke, A., D. Kendrick and E. Meeraus,

GAMS : A User's Guide(The Scientific Press, Redwood City, CA, USA, 1988).

[8] Chari, K. and T.K. Sen, An Implementation of a Graph-Based Modeling System for Structured Modeling(GBMS/SM), Decision Support Systems 22(1998), pp.103-120.

[9] CheckFree et al., Open Financial Exchange Specification 1.5, http://www.ofx.net/oft/default.asp(1998)

[10] Dolk, D.R., Model Management and Structured Modeling : The Role of an Information Resource Dictionary System, Communications of the ACM 31, No.6(1988), pp. 704-718.

[11] ebXML, "Creating a Single Global Electronic Market," http://www.ebxml.org/specindex.htm(May 2000).

[12] Ellerman, C., Channel Definition Format(CDF), http://www.w3.org/TR/NOTE-CDFsubmit.html(March 1997).

[13] Farn, C.-K. An Integrated Information System Architecture Based on Structured Modeling(Ph.D. Dissertation, University of California, Los Angeles, CA, 1985).

[14] Flammia, G., "XML and Style Sheets Promise to Make the Web More Accessible," *IEEE Expert*, May/June 1997.

[15] Fourer, R., D.M. Gay and B.W. Kernighan, AMPL : A Modeling Language for Mathematical Programming, Student Edition(The Scientific Press, Redwood City, CA, USA, 1993).

[16] Gagliardi, M. and C. Spera, BLOOMS: A Prototype Modeling Language with Object-Oriented Features, Decision Support Systems 19(1997), pp.1-21.

[17] Geoffrion, A.M., An Introduction of Structured Modeling, Management Science 33, No.5(1987), pp.547-588.

[18] Geoffrion, A.M., Computer-based Modeling Environments, European Journal of Operational Research 41(1989), pp.33-43.

[19] Geoffrion, A.M., SML : A Model Definition Language for Structured Modeling: Level 1 and 2, Operations Research 40, No.1(1992), pp.38-57.

[20] Geoffrion, A.M., SML: A Model Definition Language for Structured Modeling: Level 3 and 4, Operations Research 40, No.1,(1992), pp.58-75.

[21] Hamacher, S., Modeling Systems for Operations Research Problems: Study and Applications. Ph.D. Dissertation, Industrial Engineering, Ecole Paris Centrale, Paris, 1995.

[22] Hoff A., H. Partovi and T. Thai, Open Software Description Format(OSD), http://www.w3.org/TR/NOTE-OSD(August 1997).

[23] Huh, S.Y., Modelbase Construction with Object-Oriented Constructs, Decision Sciences 24, No.2(1993), pp.409-434.

[24] Ion P. and R. Miner, Mathematical Markup Language, http://www.W3.org/TR/WD-math(January 1998).

[25] ISO, Information Processing - Text and Office Systems - Standard Generalized Markup Language(SGML), Standard 8879(1986.)

[26] Jones, C.V., An Introduction to Graph-based Modeling Systems, Part I: Overview, ORSA Journal of Computing 2, No.2 (1990), pp.180-206.

[27] Khare, R. and A. Rifkin, "XML: A Door to Automated Web Applications," *IEEE Internet Computing*, July-August 1997.

[28] Kim, H.D. Metaview Approach to the Development of DSS Modeling Environments (Ph.D. Dissertation, KAIST, TaeJon, Repub-

lic of Korea, 1992).

[29] Kim, H.D., J.W. Kim, and S.J. Park, Web IME: An Web-based Integrated Modeling Environment for Multi-facetted Model Representation and Management, Forthcoming in The International Journal of Management Science.

[30] Kim, J.W., H.D. Kim and S.J. Park, "Multi-facetted Approach to Mathematical Model Representation and Management," Journal of the KORMS 23, No2(1998).

[31] Krishnan, R., PDM : A Knowledge-based Tool for Model Construction, Decision Support Systems 7, No.4(1991), pp.301-314.

[32] Kwon, O.B. and S.J. Park, RMT: A Modeling Support System for Model Reuse, Decision Support Systems 16, No.2(1996), pp.131-154.

[33] Lassila, O., Resource Description Framework (RDF), http://www.w3.org/RDF(February 1998).

[34] Layman, A., et al., XML-Data, http://www.w3.org/TR/1998/NOTE-XML-data(January 1998).

[35] Lee, J.K. and M.Y. Kim, Knowledge-assisted Optimization Model Formulation : UNIK-OPT, Decision Support Systems 13, No.2(1995), pp.111-132.

[36] Lee, J.S., Structure Frame Based Model Management System(Doctoral Dissertation, University of Penn., 1989).

[37] Lenard, M.L., Representing Models as Data, Journal of Management Information Systems 2, No.4(1986), pp.36-48.

[38] Lenard, M.L., An Object-oriented Approach to Model Management, Decision Support Systems 9(1993), pp.67-73.

[39] Maturana, S.V. Issues in the Design of Modeling Languages for Mathematical Programming, European Journal of Operational Research 72, No.2(1994), pp.243-261.

[40] MicroStrategy, MicroStrategy: A Leading Worldwide Provider of Enterprise DSS Software, http://www.strategy.com(1998).

[41] Muhanna, W.A., An Object-Oriented Framework for Model Management and DSS Development, Decision Support Systems 9, No.1(1993), pp.217-229.

[42] Murray-Rust, P., Chemical Markup Language(CML) 1.0, http://www.venus.co.uk/omf/cml(January 1997).

[43] OBI Consortium, Open Buying on the Internet(OBI) Technical Specifications Release V1.1, http://www.openbuy.org/obi/library(1998).

[44] OTP Consortium, Open Trading Protocol http://www.otp.org(1998)

[45] Park, S.J. and H.D. Kim, Constraint-Based Metaview Approach for Modeling Environment Generation, Decision Support Systems 9, No.4(1993), pp.325-248.

[46] Tauber, J., "XML After 1.0 : You Aint Seen Nothing Yet," *IEEE Internet Computing* (May-June 1999).

[47] W3C, Document Object Model(DOM) Level 1 Specification, http://www.w3.org/TR/REC-DOM-Level-1(October 1998).

[48] W3C, Synchronized Multimedia Integration Language(SMIL) 1.0 Specification, http://www.w3.org/TR/REC-smil(June 1998).

[49] W3C, "XSL Transformations(XSLT) Version 1.0," http://www.w3.org/TR/xslt(November 1999).

[50] W3C, "XML Schema Part 0 : Primer," http://www.org/TR/xmlschema-0(April 2000).

[51] XML/EDI Group, "XML/EDI," http://www.xmledi.com(1998).