Multidimensional Modeling using UML and XML[*]

Sergio Luján-Mora
Departamento de Lenguajes y Sistemas Informáticos
Universidad de Alicante (Spain)
slujan@dlsi.ua.es

Abstract

Multidimensional (MD) modeling is the foundation of data warehouses, MD databases, and On-Line Analytical Processing (OLAP) applications. In the past years, there have been some proposals for representing the main MD properties at the conceptual level providing their own notations. In this paper, we present an extension of the Unified Modeling Language (UML), by means of stereotypes, to elegantly represent main structural and dynamic MD properties at the conceptual level. Moreover, we use the eXtensible Markup Language (XML) to store MD models. Then, we apply the eXtensible Stylesheet Language Transformations (XSLT), that allow us to automatically generate HTML pages from XML documents, thereby supporting different presentations of the same MD model easily. Finally, we show how to accomplish all these tasks using Rational Rose 2000.

Keywords: analysis/design, multidimensional modeling, UML extensions, XML

1 Introduction

Multidimensional (MD) modeling is the foundation of data warehouses (DW), MD databases, and On-Line Analytical Processing (OLAP) applications. These systems provide companies with many years of historical information for the decision making process. Various approaches [5][10][12] for the conceptual design of MD systems have been proposed in the last few years to represent main MD structural and dynamic properties. These approaches provide their own graphical notations, which forces designers to learn a new specific model together with its corresponding notation for MD modeling. Furthermore, none of them has been widely accepted as a standard conceptual model for MD modeling. A complete comparison of MD data models can be found in [1].

On the other hand, the Unified Modeling Language (UML) [8][2] has been widely accepted as the standard object-oriented (OO) modeling language for modeling various aspects of software systems. Therefore, any approach using the UML will minimize the effort of developers in learning new notations or methodologies for every subsystem to be modeled. The UML is an extensible language, in the sense it provides mechanisms (stereotypes, tagged values, and constraints) that allow introducing new elements for specific domains if necessary, such as web applications, database applications, business modeling, software development processes, etc. [3][7].

In this paper, we present a UML extension for MD modeling based on an object-oriented MD modeling approach [11], as it easily and elegantly considers main MD properties at the conceptual level such as the many-to-many relationships between facts and dimensions, degenerate dimensions, multiple and alternative path classification hierarchies, and non-strict and complete hierarchies. Moreover, we use the eXtensible Markup Language (XML) [15] to store MD models. Then, we apply the eXtensible Stylesheet Language Transformations (XSLT) [14], that allow us to automatically generate HTML pages from XML documents, thereby supporting different presentations of the same MD model easily.

[*]This paper has been partly supported by the Spanish Ministery of Science and Technology, project number TIC2001-3530-C02-02.

Finally, all the proposals presented in this paper have been implemented: the UML extension has been programmed in Rational Rose to show its applicability, and the XML documents have been transformed using XSLT stylesheets.

The remainder of this paper is structured as follows: Section 2 introduces the main MD concepts such as fact, dimension, and hierarchy level that our approach comprises. Section 3 proposes the new UML extension for MD modeling. Section 4 shows how to use our MD extension in Rational Rose. Section 5 shows how we use XML and XSLT. Finally, Section 6 presents the main conclusions and introduces our future work.

2 Object-Oriented Multidimensional Modeling

In this section, we summarize how an OO MD model, based on the UML, can represent main structural MD properties at the conceptual level. Due to the lack of space, we will not present the dynamic part of the approach in this work.

Most of the MD features considered by this approach such as the many-to-many relationships between facts and dimensions, degenerate dimensions, multiple and alternative path classification hierarchies, and non-strict and complete hierarchies are misunderstood by most of the conceptual MD models. In this approach, the main structural properties of MD modeling are specified by means of a UML class diagram in which the information is clearly separated into facts and dimensions.

Facts and dimensions are represented by *fact classes* and *dimension classes*, respectively. Then, fact classes are specified as composite classes in shared aggregation relationships of n dimension classes. The flexibility of shared aggregation in the UML allows us to represent *many-to-many* relationships between facts and particular dimensions by indicating the 1..* cardinality on the dimension class role. For example, in Figure 1 (a), we can see how the fact class Sales has a many-to-many relationship with the dimension class Product and a one-to-many relationship with the dimension class Time.

By default, all measures in the fact class are considered additive. For nonadditive measures, additive rules are defined as constraints and are included in the fact class. Furthermore, derived measures can also be explicitly considered (indicated by /) and their derivation rules are placed between braces near the fact class, as shown in Figure 1 (a).

This OO approach also allows us to define identifying attributes in the fact class, by placing the constraint *{OID}* next to an attribute name. In this way we can represent *degenerate dimensions* [4][6], thereby providing other fact features in addition to the measures for analysis. For example, we could store the ticket number (ticket_number) as a degenerate dimension, as reflected in Figure 1 (a).

With respect to dimensions, every *classification hierarchy* level is specified by a class (called a *base class*). An association of classes specifies the relationships between two levels of a classification hierarchy. The only prerequisite is that these classes must define a Directed Acyclic Graph (DAG) rooted in the dimension class (constraint *{dag}* placed next to every dimension class). The DAG structure can represent both alternative path and multiple classification hierarchies. Every classification hierarchy level must have an *identifying* attribute (constraint *{OID}*) and a *descriptor* attribute[1] (constraint *{D}*). These attributes are necessary for an automatic generation process into commercial relational OLAP (ROLAP) tools, as these tools need to store these attributes in their metadata. The multiplicity *1* and *1..** defined in the target associated class role addresses the concepts of *strictness* and *non-strictness* respectively. Strictness means that an object at a hierarchy's lower level belongs to only one higher-level object (e.g., as one month can be related to more than one season, the relationship between both of them is non-strict). Moreover, defining the *{completeness}* constraint in the target associated class role addresses the completeness of a classification hierarchy (see an example in Figure 1 (b)). By completeness we mean that all members belong to one higher-class object and that object consists of those members only. For example, all the recorded seasons form a year, and all the seasons that form the year have been recorded. Our approach assumes all classification hierarchies are non-complete by default.

The *categorization of dimensions*, used to model additional features for class's subtypes, is considered by means of generalization-specialization relationships. However, only the dimension class can belong to

[1] A descriptor attribute will be used as the default label in the data analysis.

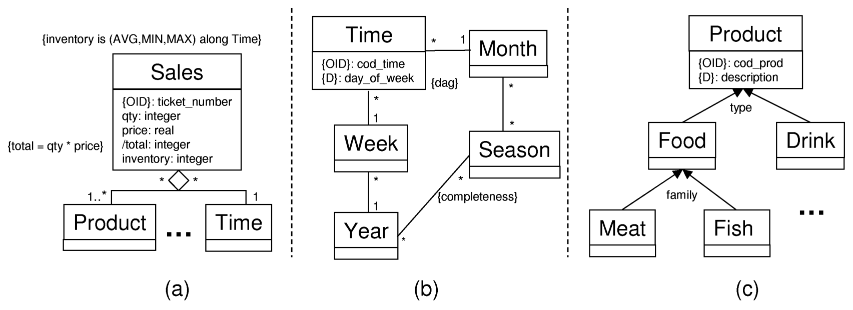


Figure 1: Multidimensional modeling using the UML

both a classification and specialization hierarchy at the same time. An example of categorization for the Product dimension can be observed in Figure 1 (c).

3 UML Extension for Multidimensional Modeling

The UML is an extensible language, in the sense it provides mechanisms (stereotypes, tagged values, and constraints) that allow introducing new elements for specific domains if necessary. In this way, the UML can be adapted and extended to fit a specific method, organization, or user. Through these extension mechanisms, a user can define and use new elements.

A *stereotype* is a model element that defines additional values (based on tagged values), additional constraints, and optionally a new graphical representation (an icon): a stereotype allows us to attach a new semantic meaning to a model element. A *tagged value* specifies a new kind of property that may be attached to a model element. Finally, a *constraint* can be attached to any model element to refine its semantics. A constraint can be defined by means of an informal explanation or by means of OCL [13][8] expressions.

We have defined eight stereotypes: three specialize the Class model element (`Fact`, `Dimension`, and `Base`), four specialize the Attribute model element (`FactAttribute`, `DimensionAttribute`, `OID`, and `Descriptor`), and one specializes the Association model element (`Completeness`). Due to the lack of space, we cannot include in this paper the complete definition of the extension[2]. Therefore, we summarize the UML elements we have used or adapted (U) and the new elements we have defined (D) to consider the main relevant MD properties:

- (D) Facts are dimensions: they are represented by means of `Fact` and `Dimension` stereotypes.
- (U) Many-to-many relationships: thanks to the flexibility of the shared-aggregation relationships, we can consider many-to-many relationships between facts and particular dimensions by means of the 1..* cardinality on the dimension class role.
- (D) Derived measures: they are represented by means of derived attributes from the UML metamodel and the tagged value `derivationRule` we have defined in the `Descriptor`, `DimensionAttribute`, and `FactAttribute` stereotypes.
- (D) Additivity: all `FactAttributes` are considered additive by default. For nonadditive `FactAttributes`, their additivity rules are defined as constraints placed near the corresponding `Fact` class.
- (U) Classification hierarchies: they are considered by means of the association between `Dimension` and `Base` stereotypes.
- (U) Strictness: the multiplicities 1 and 1..* defined in the target associated class role of a classificaton hierarchy address the concepts of strictness and nonstrictness.
- (D) Completeness: the stereotype `Completeness` addresses the completeness of a classification hierarchy.

[2]The complete definition can be found in a technical report available from `http://www.dlsi.ua.es/~slujan/files/-uml-extension.pdf`.

- **Name**: The name of the stereotype.
- **Base class** (also called Model class): The UML metamodel element that serves as the base for the stereotype.
- **Description**: An informal description with possible explanatory comments.
- **Icon**: It is possible to define a distinctive visual cue (an icon).
- **Constraints**: A list of constraints defined by means of OCL expressions associated with the stereotype, with an informal explanation of the expressions.
- **Tagged values**: A list of all tagged values that are associated with the stereotype.

Table 1: Stereotype definition schema

- (U) Categorizing dimensions: we use generalization-specialization relationships to categorize a `Dimension`.

For the definition of the stereotypes and the tagged values, we follow the structure of the examples included in the UML Specification [8]. In Table 1 we show the schema followed in our definition of the stereotypes.

For the sake of comprehensibility, we include part of the extension definition. Specifically, the following is the definition of the three stereotypes that specialize the Class model element (`Fact`, `Dimension`, and `Base`:

- Name: **Fact**
- Base class: Class
- Description: Classes of this stereotype represent facts in a MD model
- Icon: Figure 2 (a)
- Constraints:
  - All attributes of a Fact must be OID or FactAttribute:
    self.feature->select(oclIsKindOf(Attribute))->forAll(oclIsTypeOf(OID) or oclIsTypeOf(FactAttribute))
  - All associations of a Fact must be aggregations:
    self.association->forAll(aggregation = #aggregate)
  - A Fact can only be associated to Dimension classes:
    self.allOppositeAssociationEnds->forAll(participant.oclIsTypeOf(Dimension))
- Tagged values: None

- Name: **Dimension**
- Base class: Class
- Description: Classes of this stereotype represent dimensions in a MD model
- Icon: Figure 2 (b)
- Constraints:
  - All attributes of a Dimension must be OID, Descriptor, or DimensionAttribute:
    self.feature->select(oclIsKindOf(Attribute))->forAll(oclIsTypeOf(OID) or oclIsTypeOf(Descriptor) or oclIsTypeOf(DimensionAttribute))
  - All associations of a Dimension with a Fact must be aggregations at the opposite end:
    self.association.association->forAll(associationEnd.participant.oclIsTypeOf(Fact) implies associationEnd.aggregation = #aggregate)
  - All associations of a Dimension with a Fact must not be aggregations at its end:
    self.association.association->forAll(associationEnd.participant.oclIsTypeOf(Fact) implies aggregation <> #aggregate)
  - A Dimension cannot be associated to another Dimension:
    self.allOppositeAssociationEnds->forAll(not participant.oclIsTypeOf(Dimension))
- Tagged values:

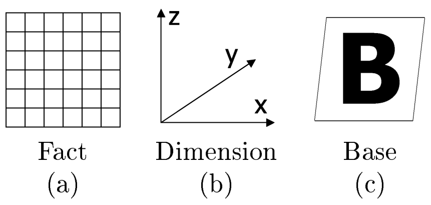


Figure 2: Stereotype icons

- isTime:
  - Type: UML::Datatypes::Boolean
  - Multiplicity: 1
  - Description: Indicates whether the dimension represents a time dimension or not

---

- Name: **Base**
- Base class: Class
- Description: Classes of this stereotype represent dimension hierarchy levels in a MD model
- Icon: Figure 2 (c)
- Constraints:
  - All attributes of a Base must be OID, Descriptor, or DimensionAttribute:
    self.feature->select(oclIsKindOf(Attribute))->forAll(oclIsTypeOf(OID) or oclIsTypeOf(Descriptor) or oclIsTypeOf(DimensionAttribute))
  - A Base must have an OID attribute and a Descriptor attribute:
    self.feature->select(oclIsKindOf(Attribute))->exist(oclIsTypeOf(OID)) and self.feature->select(oclIsKindOf(Attribute))->exist(oclIsTypeOf(Descriptor))
  - A Base can only be associated to another Base or another Dimension:
    self.allOppositeAssociationEnds->forAll(participant.oclIsTypeOf(Base) or participant.oclIsTypeOf(Dimension))
  - A Base can only be child in one generalization:
    self.generalization->size <= 1
  - A Base cannot simultaneously belong to a generalization/specialization hierarchy and an association hierarchy:
    (self.generalization->size > 0 or self.specialization->size > 0) implies (self.association->size = 0)
- Tagged values: None

4 Using Multidimensional Modeling in Rational Rose

Rational Rose (RR) is one of the most well-known visual modeling tool. RR is extensible by means of add-ins, that allow to package customizations and automation of several RR features through the Rose Extensibility Interface (REI) [9] into one component. An add-in is a collection of some combination of the following: main menu items, shortcut menu items, custom specifications, properties (UML tagged values), data types, UML stereotypes, online help, context-sensitive help, and event handling.

In this section, we present an add-in we have developed, that allows us to use the extension we have defined in RR. Therefore, we can use this tool to easily model MD conceptual models. Our add-in customizes the following elements:

- Stereotypes: We have defined the stereotypes by means of a stereotype configuration file.
- Properties: We have defined the tagged values by means of a property configuration file.
- Menu item: We have added the new menu items MD Validate and XML Export in the menu Tools by means of a menu configuration file. This first menu item runs a Rose script that validates a MD model (this script checks all the constraints defined in the UML extension for MD modeling); the second menu item runs a script that stores the MD conceptual model as an XML document.

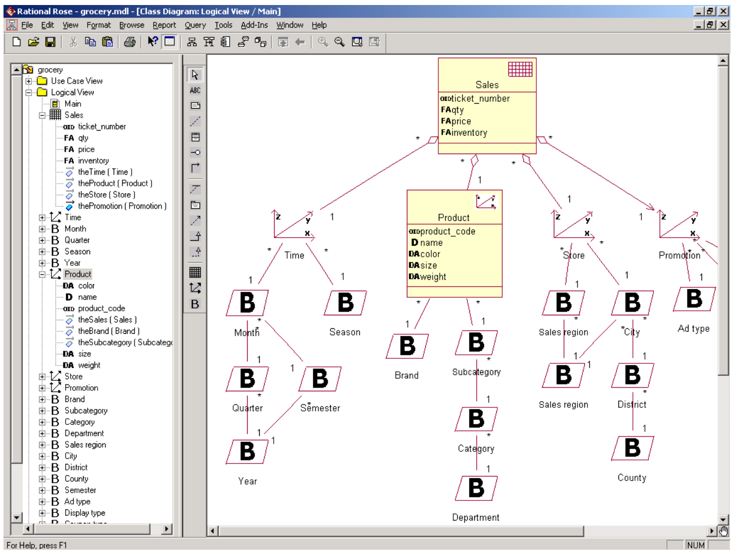


Figure 3: Multidimensional modeling using Rational Rose

The best way to understand our extension is to show a tangible example. Figure 3 shows a MD conceptual model of the well-known example "Grocery" as described in Chapter 2 of [6]. This example contains one Fact class, Sales, and four Dimension classes: Time, Product, Store, and Promotion. Every classification hierarchy level of a Dimension class is represented by a Base class. For example, the classification hierarchy of Time comprises the following Base classes: Month, Quarter, Semester, Year, and Season. For the sake of clarity, we do not show all the attributes: only the attributes of Sales and Product are displayed. We can also notice that a stereotype can be displayed in different manners in RR: Sales and Product are displayed using the stereotype icon inside the class, whereas the rest of elements are displayed using the stereotype icon.

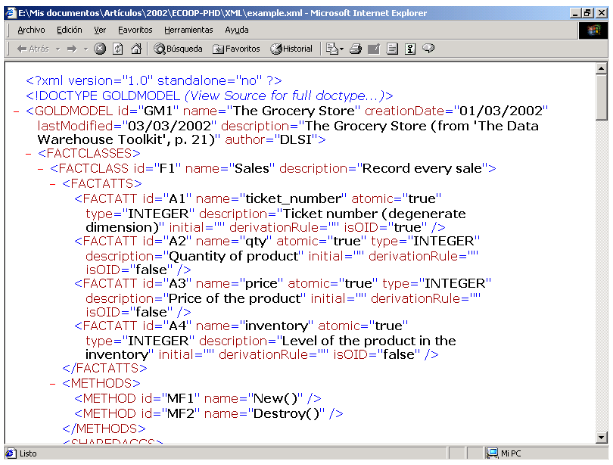
Finally, an add-in can also extend or customize RR menus by means of a menu configuration file. We have added a new menu item called MD Validate in the Tools menu. This new menu item executes a Rose script, that validates the correctness of a MD model: it checks all the OCL constraints defined in the UML extension.

5 Representing Multidimensional Models with XML

A salient issue nowadays in the scientific community and in the business world is the interchange of information. Therefore, a relevant feature of a model should be its capability to share information in an easy and standard form. The eXtensible Markup Language (XML) [15] is rapidly being adopted as a specific standard syntax for the interchange of semi-structured data. Furthermore, the XML is an open neutral platform and vendor independent meta-language standard, which allows to reduce the cost, complexity, and effort required in integrating data within and between enterprises.

We have used the XML to store MD models. The correct structure of the XML documents has been defined by means of an XML Schema. We have added a new menu item called XML Export in the Tools menu of Rational Rose. This new menu item executes a Rose script, that generates the XML documents that store instances of the MD conceptual models. Figure 4 shows the presentation of an XML document generated by our Rose script in a popular web browser (*Microsoft Internet Explorer*).

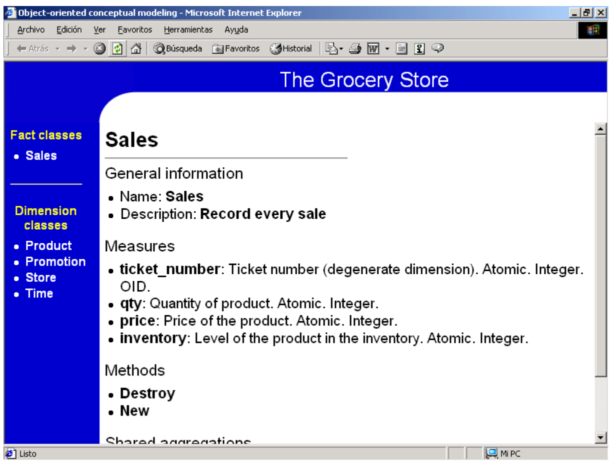
Another relevant issue of our approach was to provide different presentations of the MD models in the web at a conceptual level. However, nowadays only some web browsers partly support the XML. As a consequence, we are currently forced to transform XML documents into HTML pages in order to publish them in the web.



```
<?xml version="1.0" standalone="no" ?>
<!DOCTYPE GOLDMODEL (View Source for full doctype...)>
- <GOLDMODEL id="GM1" name="The Grocery Store" creationDate="01/03/2002"
   lastModified="03/03/2002" description="The Grocery Store (from 'The Data
   Warehouse Toolkit', p. 21)" author="DLSI">
  - <FACTCLASSES>
    - <FACTCLASS id="F1" name="Sales" description="Record every sale">
      - <FACTATTS>
          <FACTATT id="A1" name="ticket_number" atomic="true"
            type="INTEGER" description="Ticket number (degenerate
            dimension)" initial="" derivationRule="" isOID="true" />
          <FACTATT id="A2" name="qty" atomic="true" type="INTEGER"
            description="Quantity of product" initial="" derivationRule=""
            isOID="false" />
          <FACTATT id="A3" name="price" atomic="true" type="INTEGER"
            description="Price of the product" initial="" derivationRule=""
            isOID="false" />
          <FACTATT id="A4" name="inventory" atomic="true"
            type="INTEGER" description="Level of the product in the
            inventory" initial="" derivationRule="" isOID="false" />
        </FACTATTS>
      - <METHODS>
          <METHOD id="MF1" name="New()" />
          <METHOD id="MF2" name="Destroy()" />
        </METHODS>
```

E:\Mis documentos\Artículos\2002\ECOOP-PHD\XML\example.xml - Microsoft Internet Explorer
Archivo Edición Ver Favoritos Herramientas Ayuda
Atrás Búsqueda Favoritos Historial
Listo
Mi PC

Figure 4: An XML document displayed in Microsoft Internet Explorer



The Grocery Store

Fact classes
- Sales

Dimension classes
- Product
- Promotion
- Store
- Time

Sales

General information
- Name: **Sales**
- Description: **Record every sale**

Measures
- **ticket_number**: Ticket number (degenerate dimension). Atomic. Integer. OID.
- **qty**: Quantity of product. Atomic. Integer.
- **price**: Price of the product. Atomic. Integer.
- **inventory**: Level of the product in the inventory. Atomic. Integer.

Methods
- **Destroy**
- **New**

Object-oriented conceptual modeling - Microsoft Internet Explorer
Archivo Edición Ver Favoritos Herramientas Ayuda
Atrás Búsqueda Favoritos Historial
Listo
Mi PC

Figure 5: Model navigation in the web using a browser

The best method to accomplish this task is the use of the eXtensible Stylesheet Language Transformations (XSLT) [14], as a language for transforming XML documents into other XML documents. XSLT stylesheets describe a set of patterns (templates) to match both elements and attributes defined in an XML Schema, in order to apply specific transformations for each considered match. Thanks to XSLT, the source document can be filtered and reordered in constructing the resulting output. Therefore, our XML documents can be tailored to represent different presentations of the same MD model using one XSLT stylesheet for each presentation.

Our XSLT stylesheet generates a collection of HTML pages with links between them (the number of pages depends on the number of fact classes and dimension classes defined in the model). The resulting HTML pages allow us to navigate through the different presentations of the model on a web browser. All the information about the MD properties of the model is represented in the HTML pages. For example, in Figure 5 we show an example of navigation for one of the presentations. The first page is showed and contains the definition of the Sales fact class. On the left hand side, we can notice the name of the dimensions: Product, Promotion, Store, and Time. These are links that allow to navigate to the pages that contain the definition of those dimensions.

6 Conclusions and Future Work

In this paper, we have presented an extension of the UML that allows us to represent the major relevant structural MD properties at the conceptual level. This extension contains the needed stereotypes, tagged values and constraints for a complete and powerful MD modeling. We have used the OCL to specify the constraints attached to these new defined elements, thereby avoiding an arbitrary use of them. To show the applicability of this approach, we have also programmed this extension in a well-known visual modeling tool such as Rational Rose.

The main relevant advantage of this approach is that it uses the UML, a widely-accepted object-oriented modeling language, which avoids developers learning a new model and its corresponding notations for specific MD modeling.

Moreover, we have used the XML to store MD models. Then, we have applied XSLT stylesheets, that allow us to automatically generate HTML pages from XML documents, thereby easily supporting different presentations of the same MD model.

This work forms an integral part of my PhD. The aim of the PhD is to define a methodology for MD modeling based on the graphical notation introduced in this paper. This methodology will explicitly consider all underlying design guidelines that are hidden under every defined new MD element. Furthermore, we are also considering the automatic generation of database schemas into object-oriented and object-relational databases from MD conceptual models. However, this generation process is not immediate as the semantics represented by these MD conceptual models are different from those considered by the underlying MD models of OLAP tools. Therefore, it is necessary to define a generation process where some semantics represented in the conceptual model must be transformed.

Acknowledgements

I gratefully acknowledge the support of my PhD supervisor Juan Trujillo, and my colleagues Manuel Palomar and Enrique Medina, for their invaluable help in the organization of my research.

References

[1] A. Abelló, J. Samos, and F. Saltor. A Framework for the Classification and Description of Multidimensional Data Models. In *Proc. of the 12th Intl. Conference on Database and Expert Systems Applications (DEXA'01)*, pages 668–677, Munich, Germany, September 2001.

[2] G. Booch, J. Rumbaugh, and I. Jacobson. *The Unified Modeling Language: User Guide*. Object Technology Series. Addison-Wesley, 1999.

[3] J. Conallen. *Building Web Applications with UML*. Object Technology Series. Addison-Wesley, 2000.

[4] W. Giovinazzo. *Object-Oriented Data Warehouse Design. Building a star schema*. Prentice-Hall, New Jersey, USA, 2000.

[5] M. Golfarelli and S. Rizzi. A methodological Framework for Data Warehouse Design. In *Proc. of the ACM 1st Intl. Workshop on Data warehousing and OLAP (DOLAP'98)*, pages 3–9, 1998.

[6] R. Kimball. *The data warehousing toolkit*. John Wiley, 2 edition, 1996.

[7] E.J. Naiburg and R.A. Maksimchuk. *UML for Database Design*. Object Technology Series. Addison-Wesley, 2001.

[8] Object Management Group (OMG). Unified Modeling Language Specification 1.4. Internet: http://www.omg.org/cgi-bin/doc?formal/01-09-67, September 2001.

[9] Rational Software Corporation. *Using the Rose Extensibility Interface*. Rational Software Corporation, 2001.

[10] C. Sapia, M. Blaschka, G. Höfling, and B. Dinter. Extending the E/R Model for the Multidimensional Paradigm. In *Proc. of the 1st Intl. Workshop on Data Warehouse and Data Mining (DWDM'98)*, volume 1552 of *LNCS*, pages 105–116. Springer-Verlag, 1998.

[11] J. Trujillo, M. Palomar, J. Gómez, and Il-Yeol Song. Designing Data Warehouses with OO Conceptual Models. *IEEE Computer, special issue on Data Warehouses*, 34(12):66–75, 2001.

[12] N. Tryfona, F. Busborg, and J.G. Christiansen. starER: A Conceptual Model for Data Warehouse Design. In *Proc. of the ACM 2nd Intl. Workshop on Data warehousing and OLAP (DOLAP'99)*, 1999.

[13] J. Warmer and A. Kleppe. *The Object Constraint Language. Precise Modeling with UML*. Object Technology Series. Addison-Wesley, 1998.

[14] World Wide Web Consortium (W3C). XSL Transformations (XSLT) Version 1.0. Internet: http://www.w3.org/TR/1999/REC-xslt-19991116, November 1999.

[15] World Wide Web Consortium (W3C). eXtensible Markup Language (XML) 1.0 (SE). Internet: http://www.w3.org/TR/2000/REC-xml-20001006, October 2000.