

Extending UML to represent XML Schemas

Belén Vela, Esperanza Marcos

Kybele Research Group
Rey Juan Carlos University
Madrid (Spain)
{b.vela,e.marcos}@escet.urjc.es

Abstract. The consolidation of the Web as one of the most important ways to share and spread information has given rise to a huge amount of information systems for this media. This is the reason why a lot of different modelling techniques and methodologies for Web Information Systems (WISs) development have appeared. MIDAS, a model driven methodology for the development of WISs, is the framework of this paper and proposes to use UML as a unique notation to model the whole system. Despite UML does not support each of the techniques that are necessary for the development of WISs, it has some extension mechanisms that allow extending this language for other needs. In this paper a UML extension to represent XML Schemas is described.

1 Introduction

In the last decade the Web has become one of the main media for sharing and spreading information in the world. As a result, a large number of modelling techniques and methodologies for the development of Web Information Systems (WIS) [3,4,5,6,7,8,9,13,15] have appeared.

In the framework of MIDAS [10,11,12], a model driven methodology for the development of WISs, which suggests to model the whole system in a unique notation, we propose a UML extension to represent XML Schemas [16]. There are other works in this line [1,14], but these proposals have some significant limitations. The work of [1] was defined before the approval of the current standard of the World Wide Web Consortium (W3C) and was based on SOX. Therefore it does not include all the components, structures, constraints and the relationships among them of the actual XML Schema model. Moreover, it includes other components and structures that do not belong to the standard anymore. The proposal of [14] neither includes explicitly stereotypes for the components of the XML Schema model nor distinguishes among them. Besides it does not include the order of the elements and of its attributes.

One of the most significant lacks is that neither [1] nor [14] provides a unique graphical representation for the same XML Schema and vice versa; that is to say, there is no bi-directional correspondence between the graphical and the textual XML Schema representation. As a result, it is not possible to automate the XML Schema code generation in an unambiguous way, which is one of the main benefits of system modelling. For this reason, our UML extension allows to represent a XML Schema in graphical notation that has a unique correspondence with the XML Schema code.

This extension has been implemented as an add-in in Rational Rose and at this moment we are implementing the automatic XML Schema generation from a UML conceptual diagram.

In the next section the UML extension for XML Schema is described and finally, section 3 sums up the main conclusions and future work.

2 Representing XML Schemas with UML

A XML Schema [16] is the definition of a specific XML structure. A XML Schema itself is a special kind of XML document that defines the content and structure of a type of XML documents, that is, it describes the components that may be contained in a XML document and the ways the components may be arranged within in the hierarchical document structure.




XML Schema does not provide its own graphical notation, so UML will be used for this purpose. However, this language has no way of representing directly these kinds of schemas. Therefore it is necessary to extend it. UML provides an extension mechanism [2], which enables us to create new types of building blocks by means of stereotypes, tagged values and constraints, as we will see next.

The proposed UML extension is defined for the specific components of XML Schema [16], the current standard of the W3C. Each component of a XML Schema should be able to be represented in graphical notation with this UML extension, keeping the specified order and nesting.

The extension is summarized in table 1. The stereotypes have been chosen according to the following criteria:

- The ELEMENTs have been considered as stereotyped classes because they are explicitly defined in the schema. They can contain a collection of attributes.
- The attributes of an ELEMENT have been considered stereotyped attributes of the classes that represent the ELEMENT.
- The complexTypes have been considered as stereotyped classes if they are named. In this case, the complexType is related with the ELEMENT or type that uses it with a <<uses>> association. If not, they are represented in an implicit way by the compositor that it composes.
- The simpleTypes have been considered as stereotyped classes with the same name of the element that contains it. It will be related with its father ELEMENT with a composition stereotyped with <<simpleType>>.
- The complexContent types have been considered as stereotyped classes which must be related with an inheritance relationship to the father type, which must be a complexType that is redefined by the complexContent type.
- The simpleContent types have been considered as stereotyped classes be related with an inheritance relationship to the father type (simple or complex type), that is redefined by the simpleContent type.
- The compositors are considered stereotyped compositions (special kind of associations). Their stereotypes depend on the kind of compositor: <<Choice>>, <<Sequence>> and <<All>>.
- For each element, type and attribute the order number must be specified next to the class name, type name or attribute, including as a prefix the order number of its parent element or type.

Table 1. UML extension to represent in graphical notation XML Schemas

<p>ELEMENT type <i>Metamodel class:</i> Class <i>Description:</i> An <<ELEMENT>> type represents an element of the XML Schema. <i>Icon:</i>  <i>Constraints:</i> It can only be used to define <<ELEMENT>> types. <i>Tagged values:</i> The name of the element, the base type, the minimum and maximum number of occurrences. Order number with the prefix order number of the element it belongs to.</p>	<p>Attribute <i>Metamodel class:</i> Attribute <i>Description:</i> An attribute belongs to an <<ELEMENT>> type. <i>Icon:</i> None <i>Constraints:</i> An attribute can only belong to a unique <<ELEMENT>> class. <i>Tagged values:</i> The attribute name, the base type, the constraint to be satisfied by the attribute (<i>required, optional</i>) and the default or fixed value. Another tagged value is its order number including as a prefix the order number of the element to which the attribute belongs to.</p>
<p>ComplexType <i>Metamodel class:</i> Class <i>Description:</i> A <<complexType>> is a type composed of other elements or another compositor. <i>Icon:</i>  <i>Constraints:</i> It must be related by a <<uses>> association with the elements or types that use the complexType. It will only be defined as a class if it has a name, otherwise it will be defined in an implicit way. <i>Tagged values:</i> Name.</p>	<p>SimpleType <i>Metamodel class:</i> Class <i>Description:</i> A <<simpleType>> is a type that has no subelements or attributes. <i>Icon:</i>  <i>Constraints:</i> It must be associated by a <<uses>> association with the element or attribute that uses it. <i>Tagged values:</i> Base type, constraints of the proper base type.</p>
<p>ComplexContent type <i>Metamodel class:</i> Class <i>Description:</i> A <<complexContent>> is a subclass of the complexType that it defines. <i>Icon:</i> None <i>Constraints:</i> It must be related by an inheritance relationship with the elements or complexTypes that the complexContent type redefines. <i>Tagged values:</i> Name.</p>	<p>SimpleContent type <i>Metamodel class:</i> Class <i>Description:</i> A <<simpleContent>> is a subclass of the complexType or a simpleType. <i>Icon:</i> None <i>Constraints:</i> It must be related by an inheritance relationship with the type that the simpleContent type redefines. <i>Tagged values:</i> Name.</p>
<p>Association Compositor <i>Metamodel class:</i> Association <i>Description:</i> A compositor association is a special kind of composition stereotyped with the kind of compositor <<Choice>>, <<Sequence>> or <<All>>, which indicates the elements that compose the superelement (father). <i>Icon:</i> None <i>Constraints:</i> It can only be used to join an <<ELEMENT>> type with the <<ELEMENT>>s that compose it. <i>Tagged values:</i> None.</p>	<p>Association Uses <i>Metamodel class:</i> Association <i>Description:</i> A <<uses>> association is a special kind of unidirectional association which joins a named <<complexType>> with the <<ELEMENT>> or type that uses it. <i>Icon:</i> None <i>Constraints:</i> It can only be used to join to <<ELEMENT>>s or types with a named complexType. <i>Tagged values:</i> None.</p>

3 Conclusions and Future Work

In this paper we have described a UML extension to represent XML Schemas. The framework of this proposal is MIDAS, a model driven methodology for the development of WISS, which proposes to use UML as unique notation to represent the

whole system. One of the main benefits of our work is that the XML Schema graphical notation has a unique correspondence with the XML Schema code and the whole system will be defined in UML.

This extension has been implemented as an add-in in Rational Rose and at this moment we are implementing the automatic XML Schema code generation from a UML diagram.

Acknowledgements

This research is sponsored in part by the DAWIS project (TIC 2002-04050-C02-01).

References

1. Booch G., Christerson, M., Fuchs, M. and Koistinen, J., *UML for XML Schema Mapping Specification*. Retrieved from: <http://www.rational.com/media/uml/resources/media/>, 1999.
2. Booch G., Rumbaugh, J. and Jacobson I., *The Unified Modelling Language User Guide*. Addison Wesley, 1999.
3. Castano, S., Palopoli, L. and Torlone, R., *A General Methodological Framework for the Development of Web-Based Information Systems. Conceptual Modelling for E_Business and the Web*. Ed. S. W. Liddle, H. C. Mayr and B. Thalheim. Springer-Verlag, Lecture Notes in Computer Science, Berlin, 2000.
4. Conallen, J., *Building Web Applications with UML*. Addison Wesley, 2000.
5. Fraternali, P., *Tools and approaches for developing data-intensive Web applications: a survey*. ACM Computing Surveys, Vol. 31, n° 3, 1999.
6. Gómez J., Cachero C. y Pastor O., *Conceptual Modeling of Device-Independent Web Applications*. IEEE Multimedia, 8 (2), pp. 26-39, 2001.
7. Isakowitz, T., Kamis, A. and Koufaris, M., *The Extended RMM Methodology for Web Publishing*. Working Paper IS-98-18, Center for Research on Information System. Retrieved from: <http://rmm-java.stern.nyu.edu/rmm/>, 1998.
8. Koch, N., Baumeister, H. and Mandel, L., *Extending UML to Model Navigation and Presentation in Web Applications*. In Modeling Web Applications, Workshop of the UML'2000. Ed. Geri Winters and Jason Winters, York, England, October, 2000.
9. Lowe and Hall, *Hipermedia & the Web. An Engineering Approach* Ed. J. Wiley and Sons, 1999.
10. Marcos E., Vela B. and Cavero J. M., *Extending UML for Object-Relational Database Design*. Fourth Int. Conference on the Unified Modelling Language, UML 2001, Toronto (Canada), LNCS 2185, Springer Verlag, pp. 225-239, 2001.
11. Marcos, E. Vela, B., Cáceres, P. and Cavero, J.M., *MIDAS/DB: a Methodological Framework for Web Database Design*. DASWIS 2001. Yokohama (Japan), November, 2001. LNCS-2465. Springer Verlag. ISBN 3-540-44122-0. September, 2002.
12. Marcos, E. , Vela, B. and Cavero J.M. *Methodological Approach for Object-Relational Database Design using UML*. Journal on Software and System Modeling (SoSyM). Springer-Verlag. Ed.: R. France y B. Rumpe. Accepted to be published.
13. Mecca, G., Merialdo, P., Atzeni, P., and Crescenzi, V. March. *The ARANEUS guide to Web-site development*. Retrieved from: <http://poincare.inf.uniroma3.it/>, 1999.
14. Routledge, N., Bird, L. and Goodchild A. *UML and XML Schema*. ACM International Conference Proceeding Series. Proceedings of the thirteenth Australian conference on Database technologies - Volume 5, pp. 157 - 166, 2002.
15. Schwabe, D. and Rossi, G., *An object oriented approach to web-based applications design*. Theory and practice of object system, 4 (4), pp. 207-225, 1998.
16. W3C XML Schema Working Group. *XML Schema Parts 0-2: [Primer, Structures, Datatypes]*. W3C Recommendation. Retrieved from: <http://www.w3.org/TR/xmlschema-0/>, <http://www.w3.org/TR/xmlschema-1/> and <http://www.w3.org/TR/xmlschema-2/>, 2001.