Modeling Dynamic Properties in the Layered View Model for XML Using XSemantic Nets

R. Rajugan[1], Elizabeth Chang[2], Ling Feng[3], and Tharam S. Dillon[1]

[1] eXel Lab, Faculty of IT, University of Technology, Sydney, Australia
`{rajugan, tharam}@it.uts.edu.au`
[2] School of Information Systems, Curtin University of Technology, Australia
`Elizabeth.Chang@cbs.cutin.edu.au`
[3] Faculty of Computer Science, University of Twente, The Netherlands
`ling@ewi.utwente.nl`

Abstract. Due to the increasing dependence on semi-structured data, there exists a requirement to model, design, and manipulate self-describing, schema-based, semi-structured data models (e.g. XML) and the associated semantics at a higher level of abstraction than at the instance level. In this paper, we propose to model dynamic properties of a layered XML view model, at the conceptual level, using eXtensible Semantic (XSemantic) nets.

1 Introduction

Object-Oriented (OO) conceptual modeling offers the power in describing and modeling real-world data semantics and their inter-relationships in a form that is precise and comprehensible to users [1]. Conversely, XML [2] is becoming the dominant standard for storing, describing and interchanging data among various Enterprises Information Systems and databases. With the increased reliance on such self-describing, schema-based, semi-structured data language/(s), there exists a requirement to model, design, and manipulate XML data and the associated semantics at a higher level of abstraction than at the instance level.

However, existing Object-Oriented conceptual modeling languages provide insufficient modeling constructs for utilizing XML schema like data descriptions and constraints, while most semi-structured schema languages lack the ability to provide higher levels of abstraction (such as conceptual models) that are easily understood by humans. To this end, it is interesting to investigate conceptual and schema formalisms as a means of providing higher level semantics in the context of XML-related data engineering. In this paper, we use XML view as a case in point and propose to model dynamic properties of a layered XML view model [3] using eXtensible Semantic (XSemantic) nets.

In data molding, we can group the existing view models into four categories, namely [3]; (a) classical (or relational) views, (b) Object-Oriented (OO) views, (c) semi-structured (namely XML) view models [4-6, 3] and (d) view models for Semantic Web. A comprehensive discussion on existing view models can be found in our work [3]. Here, we focus only on view models for XML.

Many researchers have attempted to solve semi-structured view models by using graph based [7] and/or semi-structured data models [8, 9]. But, as in the case of relational and OO, the actual view definitions are only available at the lower levels of the implementation and not at the conceptual and/or logical level. Also, it is interesting to note that, of all the XML view models such as in [4, 5, 10, 11, 6], all provide discussion on static properties of views and only one (Active XML views [11]) provides some discussion on capturing, specifying and/or modeling dynamic nature of view properties, that have potential real-world applications in XML data engineering. In the Active XML view system, views are based on simple active rules (using non-standard, declarative ActiveView language) rather than native (XML) data or document centric view definitions. Thus, it is interesting to investigate modeling and specifying dynamic properties for XML views in a systematic manner, similar to that of describing/specifying real-world data objects in OO conceptual models.

The rest of this paper is organized as follows. In section 2 we describes our view model, followed by section 3, where we present the view modeling notation; *XSemantic nets*, with emphasis on modeling dynamic properties. Section 4 concludes the paper with some discussion on our future research directions.

2 Our Work

Our view model for XML comprised of three levels of abstraction, namely [3], *conceptual level, logical or schema level*, and *document or instance level.*

The top conceptual level describes the structure and semantics of views in a way that is more comprehensible to human users. It hides the details of view implementation and concentrates on describing objects, relationships among the objects, as well as the associated constraints upon the objects and relationships. This level can be modelled using some well-established modelling languages such as UML/OCL [12], or our own XML-specific XSemantic nets [13, 14]. The output of this level is a well-defined *valid* conceptual model in UML or XSemantic nets which can be either visual or textual (in the case of XMI models). The middle level of the view model is the schema (or logical) level describes the schema of views for the view implementation, using the XML Schema (XSD) [15] definition language. Views at the conceptual level are mapped into the view schemas at the schema level via the schemata transformation mechanism developed in previous works such as [13, 16]. The output of this level will be in either textual (XSD) or some visual (graph) notations that comply from the schema language. In our previous works such as [3, 17], we have shown how conceptual views are mapped to XSD. This includes mapping UML (view specific) stereotypes, constraints (both UML and XSemantic nets) and constructional constructs (such as bag, set, list etc.) to XSD. The third level is the document or instance level, implies a fragment of instantiated XML data, which conforms to the corresponding view schema defined at the upper level. Here, the *conceptual operators* [18] (and other view dynamic properties) are mapped to language specific query expressions (e.g. XQuery [19]), which are syntax specific.

There are two types of dynamic properties we address in our layered view model, namely; (a) view constructs: These are the sequence of one ore more conceptual operators that constructs the views and (b) internal class methods such as generic and user defined method. In this paper we address only the view constructs and the generic methods and their declarative transformation to query expression.

To illustrate our concepts in this paper, we use the description of a simple Conference Publishing System (CPS) for managing, distributing and archiving conference proceedings such as ACM, LNCS, IEEE etc. The system is similar to that of existing systems such as SpringerLink [20] or IEEE Xplore® [21].

3 Modeling Views with XSemantic Nets

The eXtensible Semantic (XSemantic) net based view design methodology comprised of three design levels: (1) semantic level, (2) schema level and (3) instance level. The aim is to enforce conceptual modeling power of semi-structured data (and views) in order to narrow the gap between real-world objects and XML document structures. The XSemantic net notation used in this paper is shown in Fig. 1.

The first level corresponds to the OO conceptual level and composes of two models, namely, the XML domain model and the XML view model. This level is based on modified semantic network [13] that provides semantic modeling of XML domains. The second level of the proposed methodology is concerned with detailed XML schema design for both domain and view objects defined at the semantic level, including *element/attribute declarations* and *simple/complex type definitions*. The mapping between these two design levels are extension of the schemata transformation proposal stated in [13] and proposed to transform the semantic models into the XML Schema, based on which XML documents can be systematically created, managed, and validated. The third level of the design methodology is concern with detailed query design for the views defined at the semantic level, including query language specific expressions and syntax declarations. The *declarative transformation* between the semantic level and the instance level are proposed to transform valid conceptual operators (and other dynamic properties) into native XML query lanague expressions, such as XQuery FLOWR expressions, Java or SQL 2003/SQLX statements. The resulting query expressions/statements are able to construct imaginary XML documents that can be validated against the XML (view) schemas, developed at the schema level of the design methodology.

The original "modified" semantic network based design methodology for XML was proposed in [13],to enforce conceptual modeling power in XML domains. It was a modified semantic net notation, to model XML domains using Object-Oriented conceptual modeling principles. The modification includes; (i) the removals of cycles in the original semantic network concept and (ii) the addition of clusters [13] (connection, connection cluster and connection cluster set) to realistically capture real-world objects and their properties or descriptions (i.e. attributes constraints etc.). By doing so, the designers can differentiate the different levels of complex and simple nodes that are used to represent real-world *objects* (e.g. PAPER) and their *properties* (e.g. PaperID, Title etc.). Later, in order to model views for XML, the "modified" semantic network was extended (called XML Semantic (XSemantic) net [14]) to include a set of conceptual operators [18] to systemically construct conceptual views from a given collection set of nodes and edges. The conceptual operators include; (i) a set of binary conceptual operators, namely union, intersection, difference Cartesian product and join and (ii) a set of unary conceptual operators, namely projection, selection, rename and restructure.

Since OO models describe both structure and behavior of an object, in order to capture dynamic properties of the views, we further extended XML Semantics nets with a new node type called ***event node***. An example of an *event node* is shown in Fig. 1-3. An *event node* is a node that describes a dynamic property (i.e. methods, messages, or triggers) associated with a complex node, using one more conceptual operators, user defined and/or generic methods (i.e. get, set, update or delete). An event node may be described as; (i) an event node *en* ε N_{ode}, is a 4-tuple, *en* = (n_{id}, n_{name}, $n_{category}$, $n_{content}$), (ii) $n_{category}$ indicating if the node is an "*event*" (this is in comparison to "*basic*" or "*complex*" categories), (iii) $n_{content}$ is a textual description of the methods. For example, the select conceptual operator node may be stated as, $n_{content} = \sigma_{paper\text{-}type=\text{“}journal\text{”}}$ and (iv) the parent of the event node is always of type complex. A complex node may have one or more simple nodes that may contain data values. Conversely, each simple node is connected to a complex node (or the root node). Thus, the event nodes associated with a complex node may be able to provide weak encapsulation for accessing and the simple nodes that are connect to the complex node in question. An XML Semantic net model with event nodes is called ***eXtensible Semantic*** (XSemantic) ***net***. A detail discussion on modeling static properties, constraints and relationships of the conceptual views can be found in [13, 17]. Here, we only discuss modeling dynamic properties using XSemantic nets.

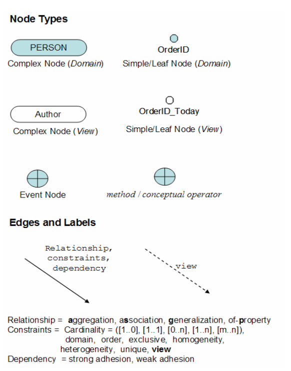
Node Types
PERSON
Complex Node (Domain)
OrderID
Simple/Leaf Node (Domain)
Author
Complex Node (View)
OrderID_Today
Simple/Leaf Node (View)
Event Node
method / conceptual operator
Edges and Labels
Relationship, constraints, dependency
view
Relationship = aggregation, association, generalization, of-property
Constraints = Cardinality = ([1..0], [1..1], [0..n], [1..n], [m..n]), domain, order, exclusive, homogeneity, heterogeneity, unique, view
Dependency = strong adhesion, weak adhesion

Fig. 1. XSemantic net notation

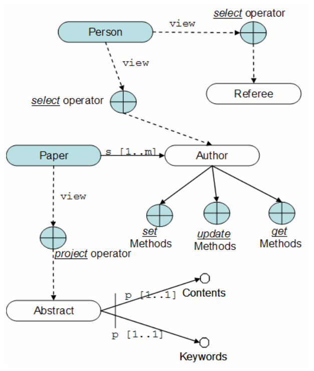

Fig. 2. Generic method examples

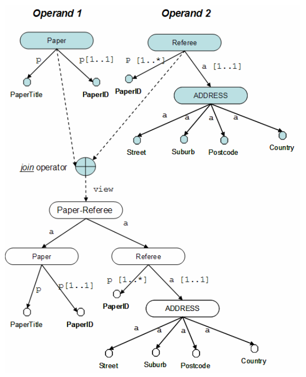

Fig. 3. JOIN operator example

Example 1: In Fig. 2, "`Referee`" is a valid XML conceptual view, named in the context of "`Person`". It is constructed using the conceptual SELECT operator, which can be shown as; $\sigma_{type=\text{“}referee\text{”}}$.

Example 2: In the case of conceptual JOIN operator with join conditions (Fig. 3), where x = `Paper` and y = `Referee/*`; $x \longrightarrow_{x.PaperId\,=\,y.paperID}\, y$.

In this paper, we use XQuery as the document view construction language and for generic methods (namely the *get* or *retrieve* methods). However, unlike SQL in relational data model, XQuery standard do not fully support XML data manipulation (for *set* and *update* operations). But, we choose XQuery as it is gaining momentum as the language of choice for XML databases and repositories, and in the future it will support many of the data manipulation features. The following examples demonstrate some of these declarative transformations.

Example 3: To transform the *get* method in the person node (Fig. 2), the following code return a person's first name; `doc ("person.xml")//firstName`.

Example 4: As shown in Fig. 2, the conceptual view operators of the view "`Referee`" can be mapped to the document view construct (XQuery expression) as shown below in the code segment.

```
for     $type in document ("person.xml")//type
where   $type = "referee"
return  <referee> {$role} </referee>
```

Example 5: As shown in example 2 (Fig. 3), we can show the conditional join conceptual operator can be mapped to the following XQuery expression, at the document level as;

```
for     $paper in document ("paper.xml"),
        $ref in document ("referee.xml")
where   $paper//papered = $ref//paperID
return  <join-example> {$paper} <referee> {$ref} </referee> </join-example>
```

4 Conclusion and Future Work

In this paper, we presented a modeling notation to capture dynamic properties in the layered view model using XSemantic nets. We also have shown a declarative transformation of such dynamic properties into document view (query) expressions.

For future work, some issues deserve investigation. First, the investigation of a formal mapping approach to conceptual view (dynamic) properties to query expressions and the automation (including efficient query constructs) of such transformation. Second, is the investigation into dynamic perspectives of the conceptual view formalism that can be applied to traditional data, Semantic Web and web services.

References

1. T. S. Dillon and P. L. Tan, *Object-Oriented Conceptual Modeling*: Prentice Hall, Australia, 1993.
2. W3C-XML, "XML 1.0, (http://www.w3.org/XML/)," 3 ed: The W3C Consortium, 2004.
3. R.Rajugan, E. Chang, T. S. Dillon, and L. Feng, "A Three-Layered XML View Model: A Practical Approach," 24th Int. Conf. on Conceptual Modeling (ER '05), Klagenfurt, Austria, 2005.
4. S. Abiteboul, "On Views and XML," Proc. of the eighteenth ACM PODS '99, USA, 1999.
5. S. Abiteboul, et al., "Active Views for Electronic Commerce," Proc. of Int. Conf. on VLDB, Scotland, 1999.
6. Y. B. Chen, et al., "Designing Valid XML Views," Proc. of the 21st Int. Conf. on ER '02, Tampere, Finland, 2002.

7. Y. Zhuge and H. Garcia-Molina, "Graph structured Views and Incremental Maintenance," Proc. of the 14th IEEE Conf. on Data Engineering (ICDE '98), USA, 1998.
8. S. Abiteboul, et al., "Views for Semistructured Data," Wrk.. on Management of Semistructured Data, USA, 1997.
9. H. Liefke and S. Davidson, "View Maintenance for Hierarchical Semistructured," Proc. of DaWak '00, UK, 2000.
10. S. Cluet, et al., "Views in a Large Scale XML Repository," Proc. of the 27th VLDB Conf. (VLDB '01), Italy, 2001.
11. S. Abiteboul, et al., "Active XML: A Data-Centric Perspective on Web Services," BDA, 2002.
12. OMG-UML™, "UML 2.0 Final Adopted Specification (http://www.uml.org/#UML2.0)," 2003.
13. L. Feng, E. Chang, and T. S. Dillon, "A Semantic Network-based Design Methodology for XML Documents," *ACM Transactions on Information Systems (TOIS)*, vol. 20, No 4, pp. 390 - 421, 2002.
14. R.Rajugan, et al., "Semantic Modelling of e-Solutions Using a View Formalism with Conceptual & Logical Extensions," 3rd Int. IEEE Conf. on INDIN '05, Perth, Australia, 2005.
15. W3C-XSD, "XML Schema (http://www.w3.org/XML/Schema)," vol. 2004, 2 ed: W3C, 2001.
16. L. Feng, E. Chang, and T. S. Dillon, "Schemata Transformation of Object-Oriented Conceptual Models to XML," *Int. Journal of Computer Systems Science & Engineering*, vol. 18, No. 1, pp. 45-60, 2003.
17. R.Rajugan, et al., "Alternate Representations for Visual Constraint Specification in the Layered View Model," The Int. Conf. on Information Integration and Web Based Applications & Services (iiWAS '05), Malaysia, 2005.
18. R.Rajugan, E. Chang, T. S. Dillon, and L. Feng, "A Layered View Model for XML Repositories & XML Data Warehouses," The 5th Int. Conf. on Computer and Information Technology (CIT '05), Shanghai, China, 2005.
19. W3C-XQuery, "XQuery 1.0 (http://www.w3.org/XML/Query): The World Wide Web Consortium (W3C), 2004.
20. Springer, "SpringerLink: http://www.springerlink.com," Springer, 2005.
21. IEEE, "IEEE Xplore®: http://ieeexplore.ieee.org," Rel 1.8 ed: IEEE, 2004.