Semantic Object Model and Flat XML Schema

Oviliani Yenty Yuliana¹, Suphamit Chittayasothorn²
ovi@peter.petra.ac.id, suphamit@kmitl.ac.th
1. Informatics Engineering Department, Petra Christian University
Surabaya, Indonesia
2. Faculty of Engineering, King Mongkut’s Institute of Technology
Ladkrabang, Bangkok, Thailand

Abstract

Generally, conceptual data models, such as E-R, NIAM, ORM, and UML, are used for modeling relational database schema. Another object database schema design, e.g. Semantic Object Model (SOM), is employed. We use SOM in our research work. The reasons, SOM can capture more meaning of an application environment and provides richer data structuring capabilities. Therefore, it is a powerful approach for constructing database applications.

On the other hand, the Extensible Markup Language (XML) becomes a popular data transfer language between database applications with different platform. So far, research and tools for transforming SOM into flat XML Schema still does not conduct.

The research project presented in this paper aims to: (1) assist understanding and documentation SOM conceptual data model and (2) propose an algorithm for transforming SOM into flat XML Schema.

Keywords: Semantic Object Model, conceptual schema, transformation, flat XML Schema

1. Introduction

Several conceptual data models have already used, such as E-R and UML. There is also SOM that is designed to capture more meaning of an application environment and provides richer data structuring capabilities for database applications. It conducted by [1, 2, 3, 4, 5].

The study performed by Auburn University found that as a data modeling tool, SOM is far superior to the E-R approach. 73% of participants could work faster with SOM. Moreover, 76% develop SOM easier than equivalent E-R models [6].

Nowadays, XML becomes a popular data transfer language between database applications with different platform. [9] states that an XML Schema can be hierarchical or flat. XML Schema serves as validation support and design documentation for a set of XML instance documents. In particular, XML Schema supplies the following additional features: XML syntax, rich data type, supporting for namespaces, constraints, and type derivations [7, 8].

There are several prior researches who propose algorithm to transform data model into XML Schema. First, the transformation ORM into XML Schema was conducted by [10]. Second, the created XML Schema from NIAM was conducted by [11]. Third, the transformation UML into XML Schema was conducted by [12, 13]. However, the transformation SOM into Flat XML Schema still do not addressed. In addition, the Salsa and the Tabledesigner tools still do not give a feature translation the data model into XML Schema.

The research project presented in this paper aims to: (1) assist understanding and documentation SOM conceptual data model and (2) propose an algorithm for transforming SOM into flat XML Schema. This paper is structured as follows. Section 2 describes SOM. Section 3 presents the algorithm to transform SOM into flat XML Schema. Finally, section 4 concludes the paper.

2. Semantic Object Model

SOM is based on concepts that were developed and published by [1, 2]. Researchers [2, 3, 4, 5, 14] use different symbols and notations to draw SOM. This research work uses symbols and notations that are proposed by [14].

2.1. Attributes

The objects have attributes that define their characteristics. For example, see BUILDING object in Figure 3 (a). There are three types of attributes, i.e. simple, group, and object. A simple attribute and a group attribute are also called as a non-object attribute. Simple attributes have a single element, e.g. Name. Group attributes are a composite of other single elements. For example, Address which contains Street, City, State, and Zip attributes. Object attributes are attributes that establish a relationship between semantic objects, e.g. APARTMENT.

2.2. Attribute Cardinality

Cardinalities are shown as subscripts of attributes in the format N.M, where N is the minimum cardinality and M is the maximum one. In Figure 3 (a), the minimum and maximum cardinality of Name are 1, which means that exactly one value of Name is required. Another example, the minimum cardinality of Telephone is 0 and maximum is 3, which means that the BUILDING may have maximum telephone...
numbers or may not have one. Two terms of attribute cardinality are used in SOM, i.e. single-value and multi-value. A single-value attribute is an attribute whose maximum cardinality is one. A multi-value attribute is one whose maximum cardinality is greater than one.

2.3. Object Identifiers

An object identifier is one or more attribute that employ to identify object instances. It is symbolized by underline ID. In BUILDING, for example, identifier is Name. A group identifier is an identifier that has more than one attribute. An example, see Figure 6 (a), ID of ASSIGNMENT is AssignmentID which consists of PROJECT and ARCHITECT object attributes.

2.4. Types of Objects

Simple object is an object that contains only single-value of simple or group attributes but no object attributes. For example, see BUILDING object in Figure 1 (a).

Composite object is an object that contains one or more multi-value of simple or group attributes but no object attributes. For instance, BUILDING object has a multi-value Telephone simple attribute, see Figure 2 (a).

Archetype/Version object produces other semantic objects that represent versions, releases, or editions of the archetype. For instance, an APARTMENT object represents a release BUILDING object, see Figure 3 (a).

Parent/Subtype object has an important characteristic called inheritance. A subtype acquires or inherits all of the attributes of its parent. For example, an EMPLOYEE object represents a parent of SUPERVISOR object and ARCHITECT object, see Figure 4 (a). A notation ST means a subtype. All relationships between the parent and the subtype are 1:1.

Compound object contains at least one object attribute. Figure 5 (a) shows a compound object between APARTMENT and PROJECT objects. APARTMENT consists of a multi-value PROJECT object attribute. On the other hand, PROJECT contains a single-value APARTMENT object attribute. This means that relationship between APARTMENT and PROJECT is one-to-many relationship.

Association object is an object that relates at least two objects and stores data that are peculiar to that relationship. For example, ASSIGNMENT object is an association object of PROJECT and ARCHITECT, as shown in Figure 6 (a).

Hybrid object is combinations of composite and compound objects. In particular, a hybrid object is an object with at least one multi-value group attribute that includes an object attribute. For instance, see Figure 7 (a). A MATERIAL-REQUISITION object has a multi-value LineItem group attribute that includes a MATERIAL object attribute as ID. This means that a particular object of MATERIAL can appear in only one of the group instances within a MATERIAL-REQUISITION.

3. Transformation SOM into Flat XML Schema Algorithm

3.1. Simple Object

There are four steps to transform simple object into flat XML Schema. First, an object is transformed into a complex type. For instance, see Figure 1, a BUILDING object is transformed into a BUILDING complex type. Second, simple attributes become simple types in XML Schema. For example, the Name and Owner simple attributes becomes the Name and the Owner simple types respectively. [7] defines the default value for both the minOccurs and the maxOccurs attributes is 1. The minimum cardinality of Owner attribute is 0 so the minOccurs="0" must be written down. Third, every a simple attribute within a single-value group becomes a simple type. For instance, Address group attribute becomes Street, City, State, and Zip simple types. At last, a key attribute becomes a key element. For example, the identifier attribute of BUILDING object is Name. It is declared in the BUILDINGKey key element.

```
1 <xs:element name="BUILDING">
2   <xs:complexType>
3     <xs:sequence>
4       <xs:element name="Name" type="xs:string"/>
5       <xs:element name="Owner" type="xs:string" minOccurs="0"/>
6       <xs:element name="Street" type="xs:string"/>
7       <xs:element name="City" type="xs:string"/>
8       <xs:element name="State" type="xs:string"/>
9       <xs:element name="Zip" type="xs:string"/>
10     </xs:sequence>
11   </xs:complexType>
12 </xs:element>
13 <xs:element name="BUILDINGID">
14   <xs:complexType>
15     <xs:sequence>
16       <xs:element name="ID" type="xs:string"/>
17     </xs:sequence>
18   </xs:complexType>
19 </xs:element>
```

Figure 1. An Example of Transformation Simple Object into Complex Type
3.2 Composite Object

There are three steps in an algorithm for transforming composite object to flat XML Schema. First, define another complex type for each multi-value simple or group attributes. If an attribute is simple than define a complex type with a new name and the attribute become a simple type of the new complex type. However, if an attribute is group than define a new complex type with the group name and the simple attributes within the group become simple types of the new complex type, for instance see Figure 2. The BUILDING object contains one multi-value simple attribute, i.e. Telephone with minimum cardinality 0 and maximum cardinality 3. The multi-value Telephone simple attribute is represented by a new PHONE complex type. Telephone and Name simple types are placed in the PHONE complex type. Second, a key in the new complex type is a compound of an identifier of the object plus a multi-value simple attribute or a compound of an identifier of the object and an identifier of the multi-value group attribute. The identifier of the BUILDING is Name. Therefore, the key of the PHONE complex type is the compound of Name and Telephone simple types. It is declared in the PHONEKey key element. The last step, to enforce referential integrity the PHONE complex type to the BUILDING complex type is defined by BUILDING-PHONE keyref element.

![Figure 2. An Example of Transformation Composite Object into Complex Type](image)

3.3. Archetype/Version Object

There are three steps for transforming composite object to flat XML Schema. First, transform an ID group attribute similar as transforming group attribute, see Section 3.2. However, an object attribute in the group attribute is transformed by the key attribute of the versioned object. It becomes a simple type. For instance, the identifier attribute of BUILDING object is Name so place Name and ApartmentNo simple types in APARTMENT complex type, see Figure 3. Second, an ID group attribute is transformed to a composite key of the archetype complex type. For instance, an ID group attribute, ApartementID, is declared in the APARTMENTKey key element with Name and ApartmentNo fields. The last step, to enforce referential integrity between the APARTMENT and the BUILDING complex type, define a BUILDING-APARTMENT keyref as in Figure 3.

3.4. Parent/ Subtype Object

There are three steps in a transforming parent/subtype object into flat XML Schema. First, declare the key attributes parent in subtype complex types as simple types. For instance, the key attribute of EMPLOYEE complex type is EmployeeNo. Therefore, place EmployeeNo simple type in SUPERVISOR and ARCHITECT complex types, see Figure 4.

Second, define key element for each subtype complex type. For example, key definition for SUPERVISOR complex types is SUPERVISORKey. In addition, key definition for ARCHITECT complex types is ARCHITECTKey. The last step, define enforce referential integrity between parent complex type and subtype complex types in keyref element. To enforce referential integrity between EMPLOYEE and SUPERVISOR complex types, define EMPLOYEE-SUPERVISOR keyref, as shown in Figure 4. As well, EMPLOYEE-ARCHITECT keyref is defined to enforce referential integrity between EMPLOYEE and ARCHITECT complex types.

3.5. Compound Object

There are three steps for transforming compound object into flat XML Schema, i.e. the step for: transforming compound objects into complex types, defining foreign key in the complex type, and defining enforce referential integrity between complex types. The algorithm for transforming the
The algorithm for transforming the compound objects that have an 1:N relationship, define a complex type for each object and place the key attribute of the parent in the child complex type. Finally, define enforce referential integrity between the parent complex type and child complex type.

The algorithm for transforming compound objects that have an N:M relationship, define a complex type for every object and additional complex type for the intersection complex type. Note that the attributes of the additional complex type are only key attribute of the complex types that are intersected. For N:M compound objects, the intersection complex type never contains non key attributes. The importance of this statement will become clear when contrast N:M compound relationship with association relationships, see Section 3.6. At last, define enforce referential integrity between the complex types and the intersection complex type.

The study case in our discussion, the relationship between APARTMENT and PROJECT objects are 1:N, see Figure 5. The APARTMENT object is transformed in Section 3.3, the result is shown in Figure 3. In addition, the transformed PROJECT object is shown in Figure 5. The APARTMENT object contains many the PROJECT object, it is shown by maximum cardinality PROJECT object attribute in the APARTMENT is N. On the other hand, the maximum cardinality APARTMENT object attribute in the PROJECT is one. Therefore, define simple types of the APARTMENT key attributes, i.e. Name and ApartmentNo. Than, declare PROJECTKey key element and define enforce referential integrity between APARTMENT and PROJECT complex type by declaring APARTMENT-PROJECT keyref.

### 3.6. Association Object

There are five steps to transform association objects in to flat XML Schema. First, transform association object structures into complex types by defining one complex type for each of the objects participating in the relationship. In Figure 6, the ASSIGNMENT object associates ARCHITECT and PROJECT objects and also stores data about their association. In this case, define three complex types,
see Section 3.1. The ARCHITECT complex type is declared in Figure 4 and the PROJECT complex type is declared in Figure 5. In addition, Figure 6 (b) shows the ASSIGNMENT complex type. As a note, the ARCHITECT and the PROJECT objects are identified as the parent objects. Second, declare key element for every parent complex types. The ARCHITECT key element is declared in Figure 4. Moreover, the PROJECT key element is declared in Figure 5. Third, the key of each parent complex types place as foreign key attributes in the complex type representing the association object. For our case, declare simple type ProjectID and EmployeeNo respectively in Figure 6. Fourth, define key element for the association object has no unique identifying attribute, the combination of the key attributes of parent complex type is used as key attributes in the association complex type. For instance, define ASSIGNMENTKey element in Figure 6 with fields ProjectID and EmployeeNo. At last, declare keyref element to enforce referential integrity among the parent complex types and the association complex type, e.g. see Figure 6.
3.7. Hybrid Object

There are four steps to transform hybrid objects to flat XML Schema. First, define a complex type for hybrid object itself, see the algorithm in Section 3.1. Second, define a complex type for every multi-value group attribute as discussion in Section 3.2.
composite object. Nevertheless, in Section 3.2, attributes within a group attribute are simple attributes. Therefore, in this section we focus on how to transform object attribute in a group attribute.

There are two possibilities object attribute in a group attribute, i.e. object attribute as a key attribute and another as a non key attribute. Key attribute of object attribute is defined as a simple type in the complex type that represents the multi-value group attribute. Third, if an object attribute is a key attribute than the defining simple type and simple type as the key attribute of hybrid object become keys on the defining complex type that represents multi-value group attribute. On the other hand, if an object attribute is a non key attribute than defining simple type as foreign in the defining complex type. In addition, the simple attribute of multi-value group attribute and the key attribute hybrid object become keys on the defining complex type. The last step, define identity constraint among hybrid object, multi-value group attribute, and object as reference object attribute complex types. Consider the MATERIAL-REQUISITION object in Figure 7 (a). It has a multi-value LINEITEM group attribute. It has a MATERIAL object attribute as a key attribute. First, define a MATERIAL-REQUISITION complex type for the object itself, the result in Figure 7 (b). Second, define LINEITEM complex type to represent multi-value LINEITEM group attribute. Third, declare key element for LINEITEM complex type. The last step, define identity constraint between MATERIAL-REQUISITION and LINEITEM. Moreover, define identity constraint between MATERIAL and LINEITEM, the result in Figure 7 (b).

4. Conclusion

SOM’s rich data structure can be transformed into flat XML Schema. In short, the steps for transforming SOM into flat XML Schema are (1) A Semantic Object and a multi-value attribute (simple or group) are transformed into complex type, see Section 3.1 and Section 3.2. (2) Single value of simple attribute is transformed into simple types, see Section 3.1. (3) Single value of group attribute is transformed into several simple types as the number of attributes within the group, see Section 3.1. (4) Single value of object attribute is used for relationship among objects. Therefore, object attribute is transformed by key attribute of the referencing object (maximum cardinality one) as simple type (foreign key) in complex type where the object attribute exist, see Section 3.5. (5) There are three kind key attributes in SOM, i.e., simple attribute, object attribute, and group attribute (of simple attributes, object attributes, or composite of simple and object attributes). If key attribute is simple attribute than simple attribute become field xpath of key element. If key attribute is object attribute than key attribute of the referencing object become field xpath of key element.

5. References

Figure 7. An Example of Transformation Hybrid Object into Complex Types