ABSTRACT: With the rapid development of the Internet, XML (Extensible Markup Language) is increasingly gaining acceptance as a medium for integrating and exchanging data. Meanwhile, object-oriented database has a strong ability to store data, XML can benefit greatly and specifically from database support and object-oriented database management system. Consequently, it is significant to exchange data from object-oriented database to XML. In particular, taking spatiotemporal information existing in many practical applications into consideration, we study the spatiotemporal data models and the problem of exchanging spatiotemporal data from object-oriented database to XML. In order to find an XML Schema which is the best description of the existing object-oriented database, we propose a comprehensive approach to focus on a set of mapping functions to accomplish the transformation of spatiotemporal data.

1. Introduction

In the current computer science field, the need of interoperability of autonomous databases results in the multi-database system, which contain heterogeneous or homogenous database. Therefore, transformation among operations of databases with different data models is critical in any heterogeneous multi-database system [18]. A mount of previous researches have focused on the problem of data exchange in different databases, for example, Fong [2, 4], Soutou [16] and Zhang and Fong [19]. With the web-based applications being more and more popular in recent years, XML which is the lingua franca for data exchange on the Internet, has become the de-facto standard of information exchange and coding format [5, 8, 12] XML. Especially, the development of XML Schema description such as DTD(Document Type Declaration) [17] and XML Schema [13] provides new opportunities for the effective data representation and handing XML data. But these advantages also bring limitation to present its structure using semantics to store data [3]. Object-oriented databases are considered better than other databases, due to the demand of new approaches to deal with complex data and complex relationship. At the same time, object-oriented databases own a strong ability to store data [9, 15]. Consequently, the research of data exchange between object-oriented database and XML is critical. There are a lot of efforts in the area, for example, the both-way transformation approach introduced in [11], the formal transformation from fuzzy objected-oriented databases to...
fuzzy XML in [7], and formal approach for reengineering data between XML and object-oriented database in [6].

In recent years, spatial [1] and temporal [14] information plays an important role in many applications such as the monitoring of environmental changes and the tracking of moving vehicle [10]. The advent of XML can provide an opportunity for the effective representation, and the advantages of object-oriented database can cope with the complex spatiotemporal data. Establishing the bridge of spatiotemporal data between object-oriented database and XML is an important research project. However, the existing methods are used to solve the common and static data, which are not suitable to copy with the multidimensional and dynamic spatiotemporal data.

In this paper, we study the methodology of transforming spatiotemporal data from object-oriented schema to XML Schema. The contributions of this paper can be summarized as follows:

• Establishing the representation of spatiotemporal data in XML and object-oriented database.
• Developing a set of functions to accomplish the transformation process of spatiotemporal data from object-oriented database to XML.

The rest of the paper is organized as follows. Section 2 firstly gives the object-oriented spatiotemporal data model. Then the transformation functions, mapping algorithm and transformation example are introduced in Section 3. Finally, Section 4 concludes the paper.

2. Object-oriented spatiotemporal data model

In this section, the object-oriented spatiotemporal data model (OOSDM) is introduced. Object-oriented spatiotemporal data have a set of characteristics which are different from the more familiar lists and table of alphanumeric data used in traditional business applications.

According to the nature of spatiotemporal data, we give the definition of object-oriented spatiotemporal data.

**Definition 1.** Object-oriented spatiotemporal data. Object-oriented spatiotemporal data is 4-tuple, STDM = (OID, AT, SP, TM), where

• OID provides a means to refer to different spatiotemporal objects.
• AT describes the non-spatiotemporal properties of spatiotemporal objects.
• SP describes the spatial properties of spatiotemporal objects.
• TM describes the temporal properties of spatiotemporal objects.

According to the definition above, we talk about all of the OID, AT, SP, TM in the following.

OID is the unique identity of spatiotemporal object. Changes of OID depict spatiotemporal objects turn into other spatiotemporal objects. It can not only represent changing type of objects (i.e., creation, split, mergence, and elimination), but also represent objects that come from objects (i.e., predecessor) and change into objects (i.e., successor).

AT is the static attributes (i.e., textual and numerical) of spatiotemporal objects, rather than dynamic attributes. Maybe there is one or more attributes in the spatiotemporal data.

SP is the spatial attributes of spatiotemporal objects. Spatial attributes include spatial position and spatial motion. There are three conditions in spatial position, which are point, line, and region. In addition, spatial motion contains motion direction and motion value. Spatial position contains point, line, and region. For simplicity, we talk about only two-dimensional regions without holes. As to point, we can use coordinate point (x, y) for representation. Fig. 1(a) shows the spatial point. Fig. 1(b) shows the line, which is represented by two coordinate points, (x1, y1) and (x2, y2), obviously, considered the straight line or approximate straight line. For spatial region, we use a series of anticlockwise points to represent it. At first, some special points are labeled, for example, (x1, y1), (x2, y2), ..., (xn, yn). Then we choose a starting point, if x ≤ min{x1, x2, ..., xn} and y ≤ min{y1, y2, ..., yn}, the point(x, y) is assumed.
as the starting point. Last, all of the labeled point are recorded in anticlockwise order, just as Fig. 1(c) shows the region, \((x_1, y_1), (x_2, y_2), \ldots, (x_5, y_5)\). Spatial motion contains direction of movement and value of movement. Certainly, we only consider the two-dimensional area.

TM is the temporal attributes of spatiotemporal objects. It includes time point and time interval. Firstly, a suitable chronon should be created. Then, time point is a chronon, and time interval is some continuous chronons. Fig. 2(a) shows the crisp time point \(I_1\), and Fig. 2(b) shows the time interval \(H_1\), from \(I_1\) to \(I_3\), which is consisted of some chronons.

![Diagram](image)

**Figure 2.** (a) Time Point, (b) Time Interval

3. Transforming spatiotemporal data from object-oriented database to XML

OOSDM schema is often represented as a class definition, and XML Schema is represented as a tree definition. The schema of transforming spatiotemporal data from OOSDM to XML maps a class-based definition to a tree-base schema definition. We assume an OOSDM schema \(F\), which includes class, object identifier (OID), reference attributes, temporal attributes, spatial attributes, common attributes (non-OID and non-spatiotemporal attributes) and method. The basic mapping functions are depicted as follows.

**Function 1:** \(F_1(\text{SchemaF}) = \text{Rname};\)

\(\text{SchemaF}\) is an OOSDM schema \(F\). \(\text{Rname}\) is a root element and it is created in the corresponding XML Schema as follows:

```xml
<xsd:schema xmlns:xsd=http://www.w3.org/2001/XMLSchema
targetNamespace="targetNamespaceURI"
xmlns="targetNamespaceURI"
elementFormDefault="qualified">
  <xsd:element name="\text{Rname}\" minOccurs = "0" maxOccurs="unbounded">
    <xsd:complexType>
      <xsd:element name="\text{Rname}\" type="xsd:ID" use = "required"/>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

**Function 2:** \(F_2(\text{Cname}) = \text{Celement};\)

\(\text{Cname}\) is a class \(C\) in OOSDM schema \(F\). \(\text{Celement}\) which is the corresponding element with the same name as \(C\) is created and then put under the root element.

```xml
<xsd:element name="\text{C}\" minOccurs = "0" maxOccurs="unbounded">
  <xsd:complexType>
    <xsd:element name="\text{C}\" type="xsd:IDREF" use = "required"/>
  </xsd:complexType>
</xsd:element>
```

**Function 3:** \(F_3(\text{Oidentifier}) = \text{Oattribute};\)

\(\text{Oidentifier}\) is an Object identifier (OID) in class \(C\) of OOSDM schema \(F\). \(\text{Oattribute}\) is the corresponding attribute in XML and \(\text{Oattribute}\) is the required attribute.

```xml
<xsd:element name="\text{C}\" minOccurs = "0" maxOccurs="unbounded">
  <xsd:complexType>
    <xsd:attribute name="\text{Oattribute}\" type="xsd:ID" use = "required"/>n
  </xsd:complexType>
</xsd:element>
```

**Function 4:** \(F_4(\text{Ridentifier}) = \text{Rattribute};\)

\(\text{Ridentifier}\) is an reference attribute in class \(C\) of OOSDM schema \(F\). \(\text{Rattribute}\) which is the corresponding attribute in XML is created and the xsd:IDREF type is also required. Then \(\text{Rattribute}\) is placed as a child element of \(C\).

```xml
<xsd:element name="\text{C}\" minOccurs = "0" maxOccurs="unbounded">
  <xsd:complexType>
    <xsd:attribute name="\text{Rattribute}\" type="xsd:IDREF" use = "required"/>
  </xsd:complexType>
</xsd:element>
```

**Function 5:** \(F_5(\text{COattribute}) = \text{COelement};\)

\(\text{COattribute}\) is a common attribute (non-OID and non-spatiotemporal attribute) or method, an element with the name \(\text{COelement}\) and data type is created and then placed as a child element of \(C\).

```xml
<xsd:element name="\text{C}\" minOccurs = "0" maxOccurs="unbounded">
  <xsd:complexType>
    <xsd:element name="\text{COelement}\" type="original-definition"/>
  </xsd:complexType>
</xsd:element>
```

Functions 1-5 can accomplish the transformation of the non-spatiotemporal attributes and method from OOSDM schema to XML Schema. As to spatiotemporal data, which exist spatial and temporal attributes, thus we will produce the transformation functions applied to spatial and temporal attributes.

Spatial attribute concludes spatial point, spatial line and spatial region. The point can be represented by coordinate point \((x,y)\), for simplicity, we assume that \(x\) and \(y\) are integer. Spatial line is represented by two coordinate points, thus we use "minOccurs="2" maxOccurs="2"" to define
two points. In addition, we use "minOccurs="3" maxOccurs="unbounded"" to define greater than or equal to three points. In the following, they are transformation function of spatial attributes.

**Function 6:** \( F_6(SPattribute) = SPelement \);

\( SPattribute \) is spatial point attribute in class C of OOSDM Schema \( F \), \( SPelement \) is the corresponding element in XML. \( SPelement \) is created and placed as a child element of \( C \).

```xml
<xsd: element name="C" minOccurs="0" maxOccurs="unbounded">
  <xsd: complexType>
    <xsd: element name="SPelement" type="dataTime"/>
  </xsd: complexType>
</xsd: element>
```

**Function 7:** \( F_7(SLattribute) = SLelement \);

\( SLattribute \) is spatial line attribute in class C of OOSDM Schema \( F \), corresponding element named \( SLelement \) is created in XML and placed as a child element of \( C \).

```xml
<xsd: element name="C" minOccurs="0" maxOccurs="unbounded">
  <xsd: complexType>
    <xsd: element name="SLelement" type="dataTime"/>
  </xsd: complexType>
</xsd: element>
```

Apart from spatial attributes, there are temporal attributes in spatiotemporal data. Temporal attribute includes time point and time interval. We use \( DateTime \) Data Type to present the time type in XML. For time interval, \( timeStart \) and \( timeEnd \) are used to represent starting time and ending time respectively. Consequently, we have the following rules.

**Function 9:** \( F_9(TPattribute) = TPelement \);

\( TPattribute \) is time point attribute in class C of OOSDM schema \( F \). A corresponding element named \( TPelement \) is created and then placed as a child element of \( C \).

```xml
<xsd: element name="C" minOccurs="0" maxOccurs="unbounded">
  <xsd: complexType>
    <xsd: element name="TPelement" type="dataTime"/>
  </xsd: complexType>
</xsd: element>
```

**Function 10:** \( F_{10}(Tlattribute) = Tlelement \);

\( Tlattribute \) is a time point attribute in class C of OOSDM schema \( F \). A corresponding element named \( Tlelement \) is created and then placed as a child element of \( C \).

```xml
<xsd: element name="C" minOccurs="0" maxOccurs="unbounded">
  <xsd: complexType>
    <xsd: element name="Tlelement" type="dataTime"/>
  </xsd: complexType>
</xsd: element>
```

Based on the transforming functions above, the algorithm which transforms spatiotemporal data from object-oriented
database to XML is proposed. The pseudo-code is depicted in Algorithm 1.

Algorithm 1 Transforming object-oriented database

Input: OOSDM schema $F$
Output: Corresponding XML Schema $T$

01 Create root element $R$ for $F$ by applying Function 1.

02 Get OOSDM class $C$, return generated XML element by applying Function 2.

03 Map the $OID$ by applying Function 3.

04 Map the $RID$ by applying Function 4.

05 Transform the common attribute (or method) by Function 5.

06 If there exists spatial attribute, Function 6, Function 7 and Function 8 can be used to transform the spatial attribute from OOSDM schema to XML Schema.

07 If there exists temporal attribute, Function 9 and Function 10 can be applied to map time point and time interval respectively.

To understand the steps of Algorithm 1, a supporting example will be taken in the following. Fig. 3 shows a class named LAND, in which there is a time point attribute, object identifier($OID$), reference attribute $RID$, common attribute $Name$, method named $Operator$, and spatial point attribute named $PPosition$ with a time interval attribute. For the class LAND, we firstly use Function 2 to map class. In addition, we can use Function 3 and Function 4 to map $OID$ and $RID$. Moreover, attribute $Name$ and method $Operator$ can be transformed by applying Function 5. Next step, $TIME$ is time point attribute, $PPosition$ is a spatial point attribute, and Function 9 and Function 6 are adopted to transform them respectively. Finally, we use Function 10 to map the time interval attribute WITH $TimeStart$ and $TimeEnd$.

4. Conclusion

In this paper, a methodology of transforming spatiotemporal data from object-oriented database to XML is proposed. At first, we present a spatiotemporal data model based on object-oriented database. Then, we produce a set of functions to accomplish the transformation process and an algorithm to deal with the exchange of spatiotemporal data from object-oriented database to XML. Last, an example is taken to illustrate our methodology.

Future work will center on three sides. Firstly, we will study the transformation of spatiotemporal data of the relationships such as generation, association, and aggregation. Secondly, the next work is to achieve the automatic transformation of spatiotemporal data from object-oriented database to XML without human intervention. Thirdly, it is also indispensable to research the transforming fuzzy spatiotemporal data between object-oriented database and XML, and finally make it suitable to specific applications.

Acknowledgements

The authors wish to thank the anonymous referees for their valuable comments and suggestions, which improved the technical content and the presentation of the paper. The work was supported by the National Natural Science Foundation of China (No. 61402087), the Natural Science
References


