IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received January 26, 2018, accepted February 23, 2018, date of publication February 27, 2018, date of current version March 19, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2809858

Reengineering Object-Oriented Fuzzy Spatiotemporal Data into XML

LUYI BAI[1,2], ZHIYI JIA[1], AND JIEMIN LIU[1,2]
[1]College of Information Science and Engineering, Northeastern University, Shenyang 110819, China
[2]Qinhuangdao Branch Campus, Northeastern University, Qinhuangdao 066004, China

Corresponding author: Jiemin Liu (liujm@neuq.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61402087, in part by the Natural Science Foundation of Hebei Province under Grant F2015501049, in part by the Scientific Research Fund of Hebei Education Department under Grant QN2014339, and in part by the Technology Support Project of NEU under Grant XNK2015003.

ABSTRACT With the rapid development of the Internet, XML has become the defacto standard for integrating and exchanging data. Since more and more business data are stored in object-oriented database, we study the methodology of modeling fuzzy spatiotemporal data and transforming fuzzy spatiotemporal data from object-oriented databases to XML as well. In order to allow for better and platform independent sharing of fuzzy spatiotemporal data stored in an object-oriented format, we devise a fuzzy spatiotemporal data model in object-oriented database to capture the semantics of spatiotemporal features. In particular, XML schema best describes the existing fuzzy object-oriented schema, and we investigate the transforming rules of spatiotemporal data from fuzzy object-oriented database to XML. Furthermore, an instance demonstrates the validation of our approach. Such approach of transformation can provide a significant consolidation of the interoperability of fuzzy spatiotemporal data from object-oriented databases to XML.

INDEX TERMS XML, object-oriented database, fuzzy spatiotemporal data, transformation.

I. INTRODUCTION

There is a considerable amount of spatiotemporal data in spatiotemporal applications [1], such as monitoring of environmental changes and tracking of moving vehicles. In the meantime, modeling spatiotemporal data has received much attention by researchers [2], [3]. In order to model spatiotemporal data accurately and effectively, there are two common approaches used widely [4]. The first is field-based model, which regards the real world to have attributes that vary over space as a continuous function. The second is object-oriented model, which assumes fully definable disjunctive objects.

As we all know, the information in spatiotemporal application exists a large of fuzzy data. The combination of spatiotemporal data and fuzziness is an important research, and two kinds of approaches are used in modeling fuzzy spatiotemporal data. The first approach focuses on using fuzzy set theory for modeling spatiotemporal objects and attributes [5]. The second approach combines spatial and temporal attributes into one modeling framework using an object-oriented modeling approach [6].

Object-oriented modeling approach is helpful to use object-oriented database. Compared to traditional database model, the object-oriented data model provides powerful object-oriented modeling capabilities and is hereby applied for representing and processing complex data in spatiotemporal applications [7]. At the same time, the object-oriented data model can copy with complex object structures and relationships among attributes [8], [9]. However, the object-oriented database is used to stored data. In order to satisfy the need of rapid development of Web-based applications [10] such as e-commerce and telemedicine-based health-care services, hence we need other technology to copy with this issue that the fuzzy spatiotemporal data stored in object-oriented database is applied to Web-based applications.

As the next generation language of the Internet, XML is becoming an increasingly critical role. Meanwhile, XML has been gradually accepted as the medium of data exchange and integration due to its simplicity, readability, and portability. The fuzzy spatiotemporal data in XML model has more advantages than traditional databases, such as self-defined tags [11], [12] and massive data type [13]. Graurav and Alhajj [11] define new tags to introduce fuzziness using both possibility theory and similarity relations, which could be in favor of determining the relationships among different fuzzy spatiotemporal data. Rizzolo and Vaisman [12] deal with temporal information by adding labeled edged with time interval as time dimensions, which could be useful to manage spatial information. In addition, the development of schema descriptions, such as DTD (Document Type Declaring) [14], and XML Schema [15], [16], opens new opportunities for an effective data representation, reasoning, and querying of XML data. Oliboni and Pozzani [13] apply a rich set of data types to define complex data types of aggregation and inheritance. In this sense, an XML Schema is more suitable for structuring complex fuzzy spatiotemporal data.

In the current computer science field, the requirement of interoperability of autonomous databases leads to the multi-database system, which contains heterogeneous or homogenous databases. Transformation between operations of databases with different data models is therefore critical in any heterogeneous multi-database system [17]. Concerning the advantages of storing data in object-oriented database and Web-based applications in XML, lots of previous works have been investigated the transformation and exchange of data between object-oriented database and XML databases [18]–[21]. There are more common features between the object-oriented model and XML, such as coping with complex data and relationships [7], [13], thus the way of mapping from object-oriented databases to XML should be less problematic. Naser *et al.* [18] propose object graph which is derived from characteristics of the schema and simply summarizes and includes all nesting and inheritance links based on XML Schema to achieve the mapping from XML to object-oriented database. Using this approach facilitates platform independent exchange of the content of object oriented databases. Furthermore, Naser *et al.* [20] investigate a similar approach of object graph to simply summarize and include all complex and simple elements and the links, which are the basics of the XML schema. Then handle the transformation of data from object-oriented databases into corresponding XML. The approach makes full use of the common features between the object-oriented model and XML, and accomplishes the transformation common data and simple relationship. Terwilliger *et al.* [21] propose a framework that addresses this problem of translating XML data into objects by transforming object-based queries and updates into queries and updates on XML using flexible, declarative mappings between classes and XML schema types. However the above methods deal with common data migration between object-oriented database and XML. They lack the strength to cope with fuzzy information.

To achieve transformation of fuzzy data between object-oriented databases and XML, there have been several achievements in combination frameworks [22], [23]. Liu *et al.* [22] introduce formal approach for reengineering fuzzy XML in fuzzy object-oriented databases. Their solution is to decompose fuzzy XML instances into different kinds of fuzziness based on XML DTD, depending on the existence of a schema describing the XML data. On the other hand, in [23], formal transformation from fuzzy object-oriented databases to fuzzy XML is produced. Liu and Ma [23] develop a series of rules for classes, attribute, method, fuzziness and relationship in object-oriented databases to transform elements and attributes in XML. Though the method achieves the exchange of fuzzy data, it only analyzes the feasibility theoretically. Meanwhile, the existing methods are solved the common and static data, they are not suitable to copy with the multidimensional and dynamic fuzzy spatiotemporal data.

- The representation of fuzzy spatiotemporal data in XML and in object-oriented database.
- We introduce a set of rules to accomplish the transforming process of fuzzy spatiotemporal data from object-oriented database to XML.

II. FUZZY SPATIOTEMPORAL DATA MODEL

A. FUZZY SPATIOTEMPORAL DATA

In this subsection, fuzziness in the spatiotemporal data model and the representation model of the fuzzy spatiotemporal database is developed.

Definition 1: FSTD (Fuzzy spatiotemporal data) is five tuples. *FSTD = (OID, AT, FAT, FSP, FTM)*, where

- *OID* (object identifier) is the unique identity of spatiotemporal object, and provides a means to refer to different spatiotemporal objects.
- *AT* is the general attributes of spatiotemporal object.
- *FAT* is the fuzzy general attributes of spatiotemporal object.
- *FSP* is the fuzzy spatial attributes of spatiotemporal object.
- *FTM* is the fuzzy temporal attributes of spatiotemporal object.

In the definition above, we will talk about *OID*, *AT*, *FAT*, *FSP*, and *FTM* in the following.

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

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

FAT is the fuzzy and static attributes (i.e., textual and numerical) of spatiotemporal objects. As to the fuzziness of attributes, we use possibility distribution to denote it.

FSP includes *FSPP* (fuzzy spatial position) and *FSPM* (fuzzy spatial motion). We firstly talk about *FSPP*. Fuzzy spatial position contains fuzzy spatial point, fuzzy spatial line, and fuzzy spatial region. A fuzzy point is a point whose exact position is not determined but possible positions are known within a certain area. For the representation of fuzzy point, we can use a polygon to surround the fuzzy point. A fuzzy line is a line, the exact shape, position, or length of which is not known, but it is known in which area the line must be. The semantic of a line is a point set between two

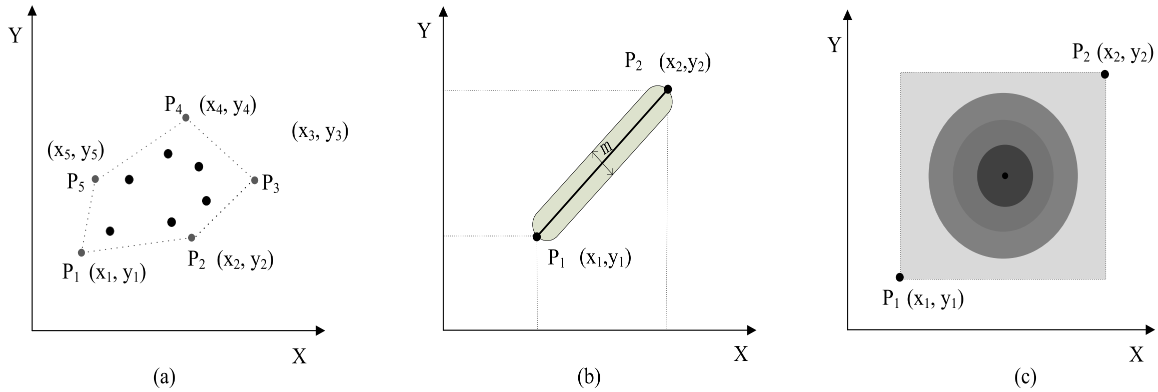


FIGURE 1. The representation of fuzzy spatial position.

ending points. A fuzzy region is a region with indeterminate boundaries.

For fuzzy spatial point, we use a series of anticlockwise points to represent it. At first, some special points are labeled, then we choose a starting point, lastly, all of the labeled points are recorded in anticlockwise order. Fig. 1(a) shows the representation of fuzzy point. The special points are $P_1(x_1, y_1)$, $P_2(x_2, y_2)$, $P_3(x_3, y_3)$, $P_4(x_4, y_4)$ and $P_5(x_5, y_5)$. Then we choose a starting point $P(x, y)$, when $x = x_{min} = min\{x_1, x_2, \ldots, x_5\}$, if only one point's x-coordinate value is equal to x_{min}, then the point (x, y) is assumed the starting point. If more than one point's x-coordinate value are equal to x_{min}, then $y = y_{min} = min\{y_1, y_2, \ldots, y_5\}$, the point (x, y) is assumed the starting point. Lastly, all of the labeled points are recorded in anticlockwise order, just as Fig. 1(a) shows the polygon, $(x_1, y_1), (x_2, y_2), \ldots, (x_5, y_5)$.

As to fuzzy spatial line, we firstly use two points to represent a straight line or approximate straight line. Then offset is used to represent the fuzzy count. Fig. 1(b) shows a fuzzy spatial line, $P_1(x_1, y_1)$ and $P_2(x_2, y_2)$ represent the straight line or approximate straight line, then the m is the offset. In reality, the fuzzy degree which is represented by function is more appropriate. For simplicity, we apply to the offset.

Fuzzy spatial region is represented by *MBR* (Minimum bounding rectangle), so that we can use two points to represent fuzzy regions. Fig. 1(c) shows the fuzzy spatial region. $P_1(x_1, y_1)$ and $P_2(x_2, y_2)$ are the two points.

FSPM is the fuzzy spatial motion, which contains fuzzy motion direction and fuzzy motion value. Fuzzy motion direction denotes that the motion is not determined and may be any. In one case, the angle of the fuzzy motion direction is in a range. In another, every motion direction owns a possible value. Fuzzy motion value actually denotes the velocity of a spatiotemporal data. The velocity is not determined and in a range.

In the following, we will discuss the representation of fuzzy motion direction. For the first condition of fuzzy motion direction, we can use two angles to represent the range of motion direction. In the coordinate axis, the positive direction of *Xaxis* is east, and the angle is 0°. In Fig. 2(a),

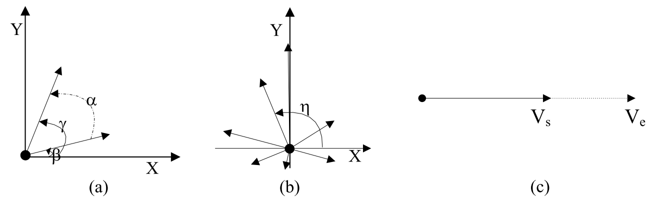


FIGURE 2. The representation of fuzzy motion direction and value.

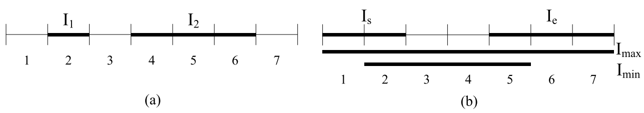


FIGURE 3. Fuzzy time points and fuzzy time interval.

β is the starting angle and γ is the ending angle. α is a fuzzy range. In Fig. 2(b), η is a possible motion direction which can be any direction, in this direction there is a fuzzy value.

As to fuzzy motion value, we use an interval to represent. Each direction owns a fuzzy motion value, like the following Fig. 2(c). The V_s is the minimum possible value, and the V_e is the maximum possible value. Consequently, the fuzzy motion value is from V_s to V_e in this direction.

FTM is the fuzzy temporal attributes of the spatiotemporal object. Fuzziness in time contains fuzziness in time point and fuzziness in time interval. The former is interpreted as a representation of a set of possible time points along with the degrees of possibility that an event occurs, and the latter is interpreted as a representation of a set of possible time intervals along with the degree of possibility.

For simplicity, we assume time as isomorphic to a natural number in this study as a result that time is linearly ordered. To describe fuzzy time, the chronon which delimit the location of the time point should be defined. Time point is a chronon, and fuzzy time point is some continuous chronons. Fuzzy time interval is consisted of two fuzzy time points. In Fig. 3(a), I_1 is the crisp chronon, and I_2 is the fuzzy time point. A fuzzy time point is described by a starting time, an ending time, and a membership function. In Fig. 3(b),

I_s and I_e denote the fuzzy starting and ending time points of the fuzzy time interval. The fuzzy time interval can be combinations of all starting time points and all ending time points. I_{max} denotes the maximum fuzzy time interval and I_{min} denotes the minimum fuzzy time interval. The membership function returns the degree to which an arbitrary chronon is part of the time interval.

B. FUZZY OBJECT-ORIENTED SPATIOTEMPORAL DATA MODEL

In this subsection, we will analyze fuzziness in the object-oriented spatiotemporal data model (*OOSDM*) and its implementation in object-oriented spatiotemporal databases, and develop the representation model of the fuzzy object-oriented spatiotemporal database.

The *OOSDM* is represented as objects, classes and relationships. In general, an object denotes an element of the domain of interest, a class denotes a set of objects with common characteristics, and a relationship denotes a relation among different classes. *OOSDM* is a class-based representation formalism that allows one to express several kinds of relationships (e.g., associations, aggregations and generations) and constraints (e.g., subclass constraints and superclass constraints) holding among the classes that are meaningful in a set of applications.

Objects have properties that may be attributes (or methods) of the object itself or relationships, between the object and one or more other objects. An object is fuzzy because of incomplete information or uncertain information. Formally, objects that have at least one attribute whose value is a fuzzy set are fuzzy objects. A class is fuzzy, because of the fuzzy domains of some attributes (or methods) fuzzy, or the subclass produced by a fuzzy class by means of specialization, and the superclass produced by some classes (in which there is at least one class who is fuzzy) by means of generalization, are also fuzzy. In addition, the relationships among classes or objects may be fuzzy. Therefore, we introduce four levels of fuzziness to the classes in *OOSDM*. These four levels of fuzziness are defined as follows:

- For the first level, classes and attribute sets of class may be fuzzy, i.e., they have a possibility to the model.
- The second level is related to the fuzzy occurrences of objects.
- The third level concerns the fuzzy values of attributes (or method) of special objects.
- The fourth level is about the relationships among classes.

In addition, there are fuzzy spatial attributes and fuzzy temporal attributes in *OOSDM*. The fuzziness of spatiotemporal attributes belongs to the third level of fuzziness. The representations of fuzzy spatiotemporal attributes are in Table 1.

In order to model the first level of fuzziness, i.e., an attribute or a class with possibility, the attribute or class name should be followed by a pair of words *WITH mem DEGREE*, where $0 \leq mem \leq 1$ is a scalar and used to indicate the degree that the attribute belongs to the class or the class belongs to the data model. As to the second level of fuzziness, we must indicate the degree of possibility that an object of the class belongs to the class. To this purpose, an additional attribute is introduced into the class to represent object membership degree to the class with attribute domain [0, 1]. We denote such a special attribute with μ in this paper. In order to model the third level of fuzziness, a keyword *FUZZY* is introduced in the model. The fourth level of fuzziness is the relationships among different classes. The relationship includes fuzzy generations, fuzzy aggregations and fuzzy association. We use the relationship membership degree with domain [0, 1] to represent fuzzy relationship.

TABLE 1. The representations of fuzzy spatiotemporal attributes.

Fuzzy spatiotemporal attribute	The representation in OOSDM
Fuzzy time point	Fuzzy *TimePoint*
Fuzzy time interval	Fuzzy *TimeInterval*
Fuzzy spatial point	Fuzzy *SpatialPoint*
Fuzzy spatial line	Fuzzy *SpatialLine*
Fuzzy spatial region	Fuzzy *SpatialRegion*
Fuzzy motion direction_1	Fuzzy *Direction_1*
Fuzzy motion direction_2	Fuzzy *Direction_2*
Fuzzy motion value	Fuzzy *MotionValue*

C. FUZZY SPATIOTEMPORAL XML DATA MODEL

In this subsection, fuzziness in the fuzzy document is discussed and the representation model of the fuzzy spatiotemporal XML database is developed.

Since XML has flexible format and the character of self-definition, XML can naturally represent fuzzy spatiotemporal information. In the case of XML, membership degrees will be associated with elements and possibility distributions will be applied to attribute values of elements. Now let us interpret what a membership degree associated with an element means. Generally, an element can nest under other elements, and these elements may have an associated membership degree. The existential membership degree associated with an element should be the possibility that the state of the world includes this element and the sub-tree rooted at it. For an element with the sub-tree rooted at it, each node in the sub-tree is not treated as independent but dependent upon its root-to-node chain. Each possibility in the source XML document is assigned conditioned on the fact that the parent element exists certainly. In other words, this possibility is a relative one based upon the assumption that the possibility the parent element exists is exactly 1.0. For attribute values of elements, we can find XML restricts attributes to a unique precise value. But in the real world, this restriction does not always hold true. It is often the case that some data items may be completely unknown and their possible values can be specified with a possibility distribution. In summary, we have two kinds of fuzziness in an XML document, the first is the fuzziness in elements, and membership degrees are used to associate with different elements. The second is the fuzziness in attribute values of elements, and we use possibility distribution to represent fuzzy values.

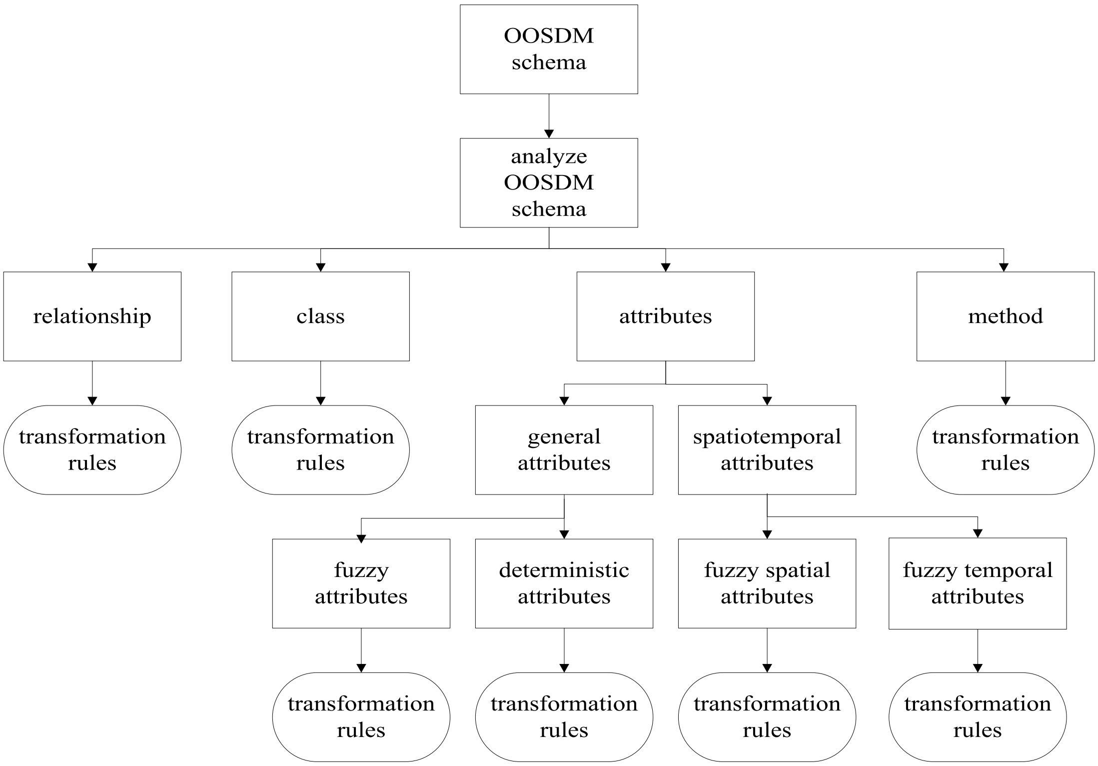


FIGURE 4. Transforming architecture.

III. TRANSFORMING FUZZY 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 translation from *OOSDM* to XML maps a class-based definition to a tree-based schema definition. There are two parts about the transformation from object-oriented database to XML. For one part, transform fuzzy spatiotemporal data from object-oriented database to flat XML (flat XML is a very simple XML format that only has simple leaf elements and attributes as subelements of non-leaf elements). For another part, transformation of fuzzy spatiotemporal data from object-oriented database to nested XML (nested XML is a comparatively complex XML format which includes the representation of relationship between two root elements). The transformation architecture fig. is Fig. 4. According to Fig. 4, what we should accomplish is to produce transformation rules which contain relationship among different classes, class and fuzzy attributes (or methods), deterministic attributes (or methods), fuzzy spatial attributes and fuzzy temporal attributes in class. All of the transformation rules will be depicted in the following subsection.

A. TRANSFORMING FUZZY SPATIOTEMPORAL DATA FROM OBJECT-ORIENTED DATABASE TO FLAT XML

We assume an *OOSDM* schema S, which includes class, object identifier (*OID*), fuzzy temporal attributes, fuzzy spatial attributes, general attributes (non-*OID* and non-spatiotemporal attributes) and methods. The general attributes and method may or may not fuzzy. According to the Fig. 4, there are many rules to be introduced and the basic mapping rules are depicted as follows. The basic mapping rules are depicted as follows.

Rule 1: For an *OOSDM* schema S, a root element named *XMLRoot* is created in the corresponding XML Schema.

Rule 2: For each class C in *OOSDM* schema S, an element with the same name as C is created and then put under the root element.

Rule 3: For each object identifier (*OID*) with the name O_i in class C of *OOSDM* schema S, the xs:ID type is used for creating an attribute of the element for C.

Rule 4: For each general attribute (or method) Ai in class C of *OOSDM* schema S, an element with corresponding name and data type is created and then placed as a child element of C.

Rules 1–4 could be directly used to process the transformation from *OOSDM* schema to XML Schema when all the classes, attributes (or methods) are deterministic. As soon as there is fuzziness in *OOSDM*, the following rules (Rules 5–7) should be used.

Rule 5: If there exists the first level of fuzziness in *OOSDM* (recall the definition of fuzziness in *OOSDM* in Sect.2.2), for each general attribute (or method) A_i in class C of *OOSDM* schema S, the following strategy is used, in particular, an element *Val* is firstly created as a child element of C, and then an element is placed as a child element of *Val* with corresponding name and data type is created.

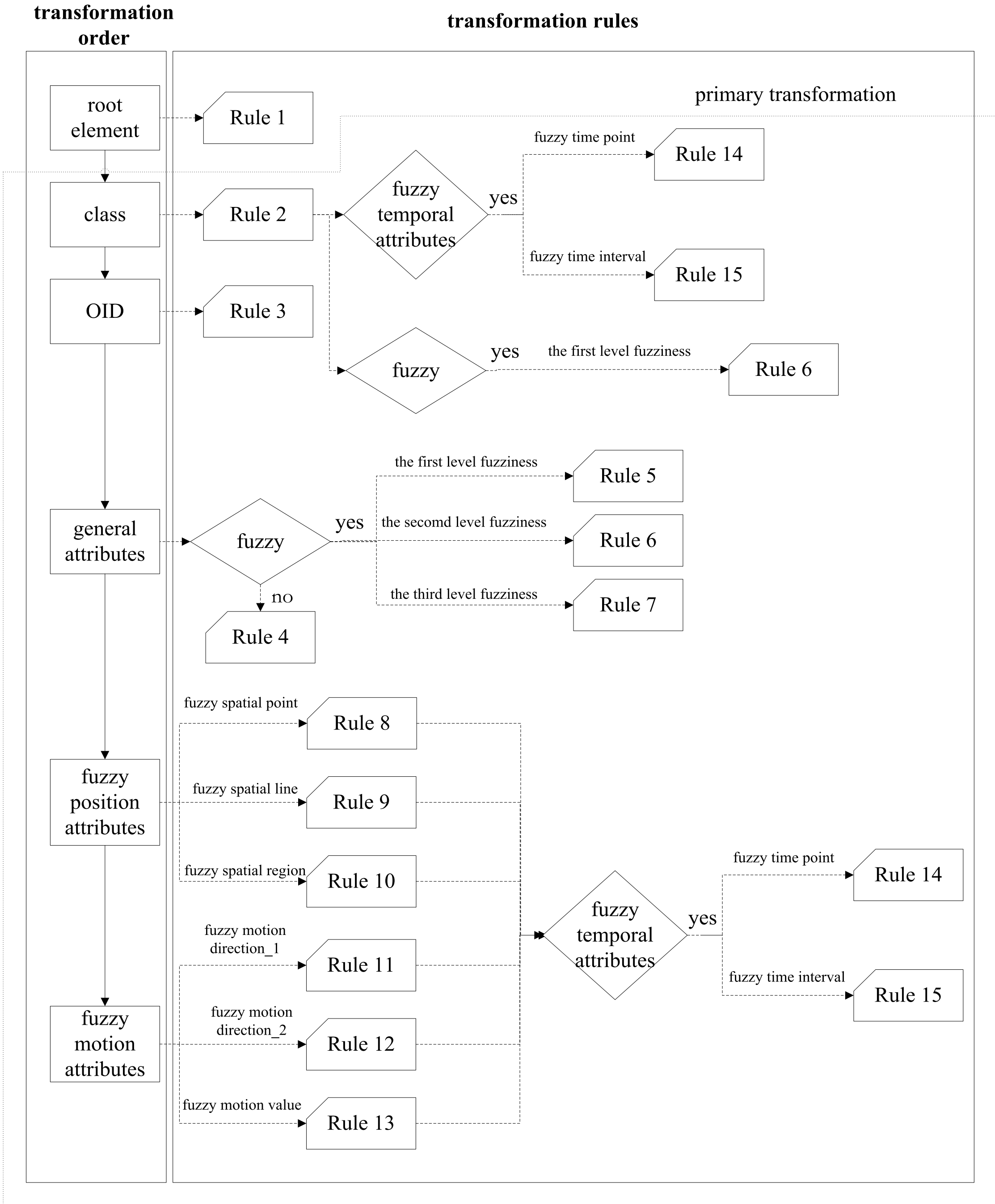


FIGURE 5. Transforming flow diagram of Algorithm 1.

Rule 6: If there exists the second level of fuzziness in *OOSDM* (recall the definition of fuzziness in *OOSDM* in Sect.2.2), for each general attribute (or method) A_i in class C of *OOSDM* schema S, the following strategy is used, in particular, an element *Dist* is firstly created as a child element of C, an element *Val* is then created as a child element of *Dist* and finally an element placed as a child element of *Val* with corresponding name and data type is created.

Rule 7: If there exists the third level of fuzziness in *OOSDM* (recall the definition of fuzziness in *OOSDM* in Sect. 2.2), for each general attribute (or method) *Ai* in class C of *OOSDM* schema S, the transforming strategy is as follows: an element *Dist* is firstly created as a child element of *Ai*, then element *Val* is created as a child element of *Dist*, and last element *Vi* which is the fuzzy value is created as a child element of *Val*.

In the above rules, we talk about the transformation of general attributes from *OOSDM* schema to XML Schema. However in the *OOSDM* there are some spatial attributes and temporal attributes. Thus, the rules of fuzzy spatiotemporal attributes will be studied.

In definition 1, we introduce the fuzzy spatial position. Consequently, the rule 8-10 will be applied to fuzzy spatial point, fuzzy spatial line and fuzzy spatial region.

Rule 8: For an *OOSDM* schema S, *SpatialPoint* is the fuzzy spatial point attributes, and an element with the same name *SpatialPoint* is created as a child element of C.

Rule 9: For an *OOSDM* schema S, *Spatialline* is the fuzzy spatial line attributes, and the element *Spatialline* in XML is created as a child element of C.

Rule 10: For an *OOSDM* schema S, *SpatialRegion* is the fuzzy spatial line attributes, and the corresponding XML Schema of an element *SpatialRegion* is created.

For fuzzy spatial attributes, apart from fuzzy spatial position, there exists fuzzy spatial motion which is includes fuzzy motion direction and fuzzy motion value. Fuzzy motion direction contains two conditions. In one case, the angle of the fuzzy motion direction is in a range. In another, every motion direction owns a possible value. Rule 11 and Rule 12 are used to map them.

Rule 11: For an *OOSDM* schema S, *Direction_1* is the first-type fuzzy motion direction attributes, an element with the same name *Direction_1* is created as a child element of C.

Rule 12: For an *OOSDM* schema S, *Direction_2* is the second-type fuzzy motion direction attributes, and the element *Direction_2* in XML is created as a child element of C.

Rule 13: For an *OOSDM* schema S, *Motionvalue* is the fuzzy motion value attributes, and the corresponding XML Schema of an element *Motionvalue* is created.

In the following, we will talk about the fuzzy temporal attributes which include fuzzy time point and fuzzy time interval.

Rule 14: For an *OOSDM* schema S, *TimePoint* is the fuzzy time point attributes, the corresponding XML Schema of an element *TimePoint* is created.

Rule 15: For an *OOSDM* schema S, *TimeInterval* is the fuzzy time interval attributes, an element with the same name *TimeInterval* is created as a child element of C.

The detailed XML Schema of Rule 1-15 is depicted in Fig. 7. Based on the transformation rules and analysis, it is easy to obtain the following transforming flow diagram of algorithm 1.

In order to understand Algorithm 1 intuitively, the following flow diagram in Fig. 5 is represented. The left column shows the transformation order and the right column shows the transformation rules. The primary transformation is in the dotted box.

B. TRANSFORMING FUZZY SPATIOTEMPORAL DATA FROM OBJECT-ORIENTED DATABASE TO NESTED XML

In this subsection, we will give the transforming rules of the relationships such as fuzzy generalization, fuzzy association and fuzzy aggregations among fuzzy *OOSDM* classes. The generalization in the *OOSDM* defines a subclass/ superclass relationship between classes: one class, called superclass,

Algorithm 1 Transforming *OOSDM* Schema to flat XML

Input: OOSDM schema *S*

Output: transformed fuzzy XML Schema *X*

01 create root element *XMLRoot* for S by applying Rule 1
02 get next fuzzy *OOSDM* class C, return generated fuzzy XML element by applying Rule 2
03 transform the *OID* by applying Rule 3
04 transform the deterministic general attributes (or method) by applying Rule 4
05 if there exists fuzziness of the in C, then map the object with the three levels of fuzziness by applying by Rule 5, 6, and 7
06 if the class C and general attributes own fuzzy temporal attributes, use Rule 14 and 15 to map them
07 for the fuzzy spatial position attributes, use Rule 8, 9, and 10 to map them, if they simultaneously own fuzzy temporal attributes, apply the Rule 14 and 15
08 for the fuzzy spatial motion attributes, by applying Rule 11, 12, and 13, if they simultaneously own fuzzy temporal attributes, use the Rule 14 and 15 to map them

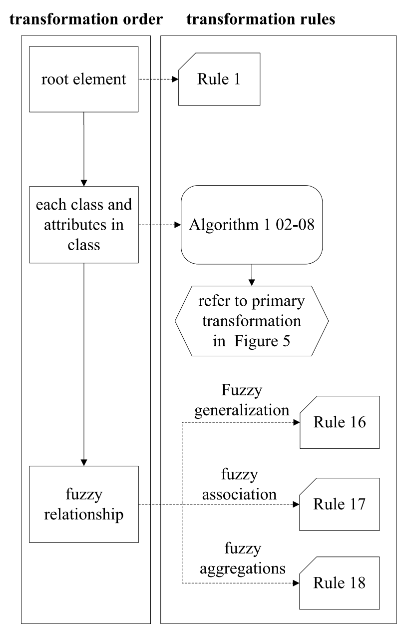


FIGURE 6. Transforming flow diagram of Algorithm 2.

is a more general description of a set of other classes, called subclasses. In particular, we have the following rule:

Rule 16: For each subclass *SC* in *OOSDM*, let its superclass be C, then an element similar to Rule 2 and with the same name as *SC* is created. In order to represent the fuzzy generation, an element named *fuzzyGeneration* is defined and this element contains two subelements which one uses "extension" tag to inherit properties of C and another one represents the degree of membership. S_i is the attribute (or method) in class *SC*.

Associations among *OOSDM* database schema are represented by using links. In practical applications, these

Rule 1
```
<?xml version="1.0"
encoding="UTF-8"?>
<xs:schema xmlns:xs="http://
www.w3.org/2001/XMLSchema"
elementFormDefault="qualified"
attributeFormDefault="unqualified"
>
 <xs:element name="XMLRoot">
  <xs:complexType>
   <xs:sequence>
    <!--transformed fuzzy XML
Schema of S -->
   </xs:sequence>
  </xs:complexType>
 </xs:element >
</xs:schema>
```

Rule 2
```
<xs:element name="C"
minOccurs="0"
maxOccurs="unbounded">
 <xs:complexType>
  <!--sub-elements of C -->
 </xs:complexType>
</xs:element>
```

Rule 3
```
<xs:element name="C"
minOccurs="0"
maxOccurs="unbounded">
 <xs:complexType>
  <xs:attribute name="Oi"
type="xs:ID" use="required"/>
 </xs:complexType>
</xs:element>
```

Rule4
```
<xs:element name="C" minOccurs
="0" maxOccurs ="unbounded">
 <xs:complexType>
  <xs:element name ="Ai" type
="original-definition"/>
 </xs:complexType>
</xs:element>
```

Rule 5
```
<xs:element name="Ai" minOccurs
="0" maxOccurs ="unbounded">
 <xs:complexType>
  <xs:element name ="Val"/>
 </xs:complexType>
</xs:element>
<xs:element name="Val" minOccurs
="0" maxOccurs ="unbounded">
 <xs:complexType>
  <xs:attribute name ="Poss" type
="xs:fuzzy" default="1.0"/>
 </xs:complexType>
</xs:element>
```

Rule 6
```
<xs:element name="C"
minOccurs="0"
maxOccurs="unbounded">
 <xs:complexType>
  <xs:element name ="Dist"/>
 </xs:complexType>
</xs:element>
<xs:complexType name="DistType"
minOccurs ="0"
maxOccurs ="unbounded">
 <xs:sequence>
  <xs:element name ="Val"/>
 </xs:sequence>
</xs:complexType>
<xs:element name="Val" minOccurs
="0" maxOccurs ="unbounded">
 <xs:complexType>
  <xs:element name ="Ai" type
="original-definition"/>
  <xs:attribute name ="Poss" type
="xs:fuzzy" default="1.0"/>
 </xs:complexType>
</xs:element>
```

Rule 7
```
<xs:element name="Ai" minOccurs ="0"
maxOccurs ="unbounded">
 <xs:complexType>
  <xs:element name ="Dist"/>
 </xs:complexType>
</xs:element>
<xs:complexType name="Dist"
minOccurs ="0"
maxOccurs="unbounded">
 <xs:sequence>
  <xs:element name ="Val"/>
 </xs:sequence>
</xs:complexType>
<xs:element name="Val" minOccurs ="0"
maxOccurs="unbounded">
 <xs:complexType>
  <xs:element name ="Vi" type
="original-definition"/>
 </xs:complexType>
</xs:element>
```

Rule 8
```
<xs:element name="SpatialPoint">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="No."
type="xs:integer"/>
   <xs:element ref="point"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
<xs:element name="point"
minOccurs="1"
maxOccurs="unbounded">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="Xaxis" type="
xs:integer"/>
   <xs:element name="Yaxis" type="
xs:integer"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
```

Rule 9
```
<xs:element name="Spatialline">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="No."
type="xs:integer"/>
   <xs:element ref="point "/>
  </xs:sequence>
  <xs:attribute name="offset"
type="xs:integer"/>
 </xs:complexType>
</xs:element><xs:element name="point"
minOccurs="2" maxOccurs="2">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="Xaxis"
type="xs:integer"/>
   <xs:element name="Yaxis"
type="xs:integer"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
```

Rule 10
```
<xs:element name="SpatialRegion">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="No."
type="xs:integer"/>
   <xs:element ref="point"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
<xs:element name="point"
minOccurs="2" maxOccurs="2">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="Xaxis"
type="xs:integer"/>
   <xs:element name="Yaxis"
type="xs:integer"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
```

Rule 11
```
<xs:element name="Direction_1">
 <xs:complexType>
  <xs:sequence>
   <xs:element ref="degree"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
<xs:element name="degree"
minOccurs="2" maxOccurs="2">
 <xs:simpleType>
  <xs:restriction base="xs:integer">
   <xs:minInclusive
value="degreeStart"/>
   <xs:maxInclusive
value="degreeEnd"/>
  </xs:restriction>
 </xs:simpleType >
</xs:element>
```

Rule 12
```
<xs:element name="Direction_2">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="degree"
type="xs:integer"/>
   <xs:element name ="Val"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
<xs:element name="Val" minOccurs
="0" maxOccurs ="unbounded">
 <xs:complexType>
  <xs:attribute name ="Poss" type
="xs:fuzzy" default="1.0"/>
 </xs:complexType>
</xs:element>
```

Rule 13
```
<xs:element name="Motionvalue">
 <xs:simpleType>
  <xs:restriction base="xs:integer">
   <xs:minInclusive
value="valueStart"/>
   <xs:maxInclusive
value="valueEnd"/>
  </xs:restriction>
 </xs:simpleType>
</xs:element>
```

Rule 14
```
<xs:element name="TimePoint ">
 <xs:restriction base="dateTime">
  <xs:minInclusive
value="valueStart"/>
  <xs:maxInclusive
value="valueEnd"/>
 </xs:restriction>
</xs:element>
```

Rule 15
```
<xs:element name="TimeInterval">
 <xs:complexType>
  <xs:sequence>
    <xs:element name="timeStart"/>
    <xs:element name="timeEnd"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
<xs:element name="timeStart"
minOccurs="0"
maxOccurs="unbounded">
 <xs:simpleType>
  <xs:restriction base="dateTime">
   <xs:minInclusive
value="valueStart"/>
   <xs:maxInclusive
value="valueEnd"/>
   </xs:restriction>
 <xs:simpleType>
</xs:element>
<xs:element name="timeEnd"
minOccurs="0"
maxOccurs="unbounded">
 <xs:simpleType>
  <xs:restriction base="dateTime">
   <xs:minInclusive
value="valueStart"/>
   <xs:maxInclusive
value="valueEnd"/>
   </xs:restriction>
 <xs:simpleType>
</xs:element>
```

Rule 16
```
<xs:element name="C" minOccurs ="0"
maxOccurs ="unbounded">
 <xs:complexType>
  <!--sub-elements of C -->
 </xs:complexType>
</xs:element>
<xs:element name="LC" minOccurs ="0"
maxOccurs ="unbounded">
 <xs:complexType>
  <xs:element name ="fuzzyAssociation"
>
   <xs:complexType>
    <xs:extension base= "C"/>
    <xs:element name="Val">
      <xs:complexType>
       <xs:attribute name ="Poss" type
="xs:fuzzy" default="1.0"/>
      </xs:complexType>
    </xs:element>
   </xs:complexType>
  </xs:element>
  <xs:element name="Si" type="original-
definition"/>
 </xs:complexType>
</xs:element>
```

Rule 17
```
<xs:element name="C" minOccurs ="0"
maxOccurs ="unbounded">
 <xs:complexType>
  <!--sub-elements of C -->
 </xs:complexType>
</xs:element>
<xs:element name="LC" minOccurs ="0"
maxOccurs ="unbounded">
 <xs:complexType>
  <xs:element name ="fuzzyAssociation"
>
   <xs:complexType>
    <xs:extension base= "C"/>
    <xs:element name="Val">
      <xs:complexType>
       <xs:attribute name ="Poss" type
="xs:fuzzy" default="1.0"/>
      </xs:complexType>
    </xs:element>
   </xs:complexType>
  </xs:element>
  <xs:element name="Si" type="original-
definition"/>
 </xs:complexType>
</xs:element>
```

Rule 18
```
<xs:element name="PCi" minOccurs ="0"
maxOccurs="unbounded">
 <xs:complexType>
  <!--sub-elements of PCi -->
 </xs:complexType>
</xs:element>
<xs:element name="WC" minOccurs ="0"
maxOccurs="unbounded">
 <xs:complexType>
  <xs:element name ="fuzzyAggregation"
>
   <xs:complexType>
    <xs:element ref="PCi"/>
    <xs:element name="Val">
      <xs:complexType>
       <xs:attribute name="Poss"
type="xs:fuzzy" default="1.0"/>
      </xs:complexType>
    </xs:element>
   </xs:complexType>
  </xs:element>
  <xs:element name="Si" type="original-
definition"/>
 </xs:complexType>
</xs:element>
```

FIGURE 7. The XML schema of rule 1-18.

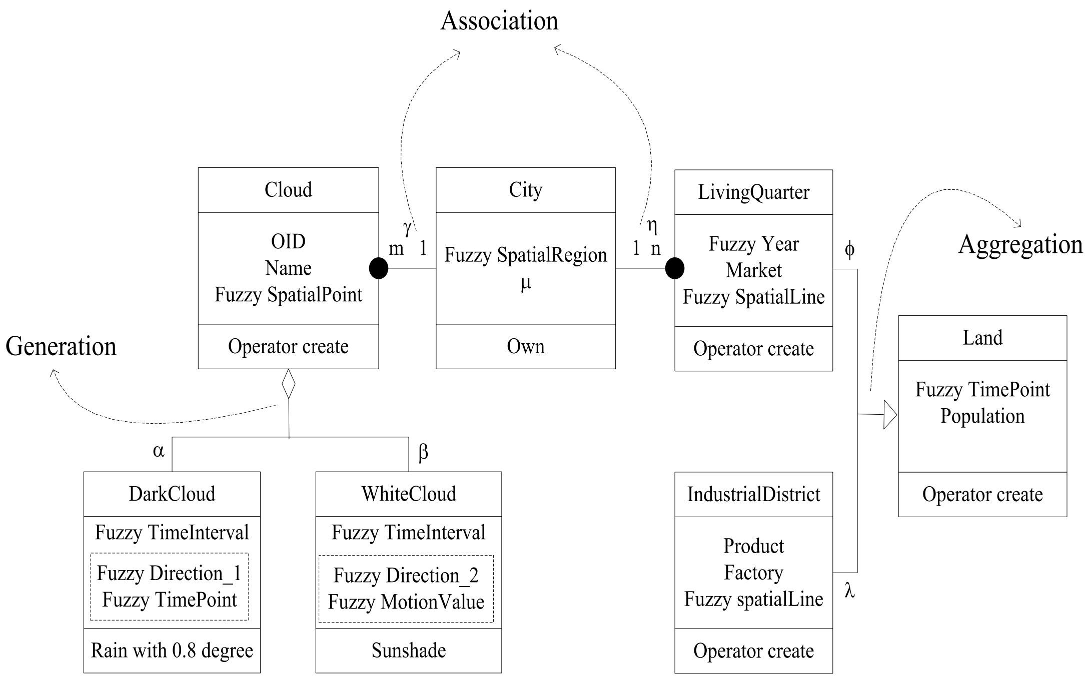


FIGURE 8. Fuzzy OOSDM database schema.

TABLE 2. The range of wind speed in P1, P2 and P3.

Class	Wind speed(mph)	
	minValue	maxValue
P1	39	73
P2	74	95
P3	96	110

TABLE 3. The range of pressure in P1, P2 and P3.

Class	Pressure (mb)	
	minValue	maxValue
P1	986	999
P2	940	973
P3	960	963

relationships could be effectively identified by class involvement with n: m. Here, n and m are the constrained by cardinality. In particular, we have the following rule:

Rule 17: For every class *C* in *OOSDM*, let its linked class be *LC*, then an element similar to Rule 2 and with the same name as *LC* is created. In order to represent the fuzzy association, an element named *fuzzyAssociation* is defined and this element contains two subelements which one uses "extension" tag to identity relationship between *C* and *LC*, and another one represents the degree of membership. For *LC*, *maxOccurs* = "m" represents the involvement 1: m and *maxOccurs* = "unbounded" represent involvement unknown. S_i is the attribute (or method) in class *LC*.

Aggregation is an abstraction through which objects in composition relationships are treated as higher-level objects. In the XML Schema, the transformation of aggregation is to group sub-elements (with respect to part-class in fuzzy *OOSDM*) under an element (with respect to whole-class in fuzzy *OOSDM*). In particular, we have the following rule:

Rule 18: For each part-class PC_i in *OOSDM*, let its whole-class be *WC*, then an element similar to Rule 2 and with the same name as *WC* is created. In order to represent the fuzzy association, an element named *fuzzyAggregation* is defined and this element con tains two subelements which one uses "ref" tag to refer to PC_i, and another one represents the degree of membership. S_i is the attribute (or method) in class *WC*.

The detailed XML Schema of Rule 16-18 is depicted in Fig. 7. Based on the transforming rules above, we now present the algorithm for transforming *OOSDM* schema to nested XML Schema. The pseudo-code is depicted in Algorithm 2.

Algorithm 2 Transforming *OOSDM* Schema to nested XML

Input: *OOSDM* schema *S*
Output: transformed nested XML Schema *X*

01 create root element *XMLRoot* for S by applying Rule 1
02 transform each class and attributes in class by applying Algorithm 1 02-08
03 for fuzzy generalization, use Rule 16 to map it.
04 for fuzzy association, use Rule 17 to map it.
05 for fuzzy aggregations, use Rule 18 to map it.

According to Algorithm 2, the flow diagram of transformation is represented in Fig. 6. In this flow diagram, we refer to the primary transformation in Fig. 5.

1. The class *City* is mapped as follow:

```
<xs:element name="City" minOccurs="0" maxOccurs="unbounded">
 <xs:complexType>
  <xs:element name="Dist"/>
 </xs:complexType>
</xs:element>
<xs:complexType name="Dist" minOccurs="0" maxOccurs="unbounded">
 <xs:sequence>
  <xs:element name ="Val"/>
 </xs:sequence>
</xs:complexType>
<xs:element name="Val" minOccurs="0" maxOccurs="unbounded">
 <xs:complexType>
  <xs:element name="Own" type ="original-definition"/>
  <xs:element name="SpatialRegion"/>
  <xs:attribute name ="Poss" type ="xs:fuzzy" default="1.0"/>
 </xs:complexType>
</xs:element>
<xs:element name="SpatialRegion" minOccurs="0" maxOccurs="unbounded">
 <xs:complexType>
  <xs:sequence>
   <xs:element ref="point "/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
<xs:element name="point" minOccurs="2" maxOccurs="2">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="Xaxis" type="xs:integer"/>
   <xs:element name="Yaxis" type="xs:integer"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
```

2. The linked and superclass *Cloud* is mapped as follow:

```
<xs:element name="Cloud" minOccurs="0" maxOccurs="unbounded">
 <xs:complexType>
  <xs:element name="fuzzyAssociation">
   <xs:complexType>
    <xs:extension base="City"/>
    <xs:element name="Val">
     <xs:complexType>
      <xs:attribute name ="Poss" type ="xs:fuzzy" default="1.0"/>
     </xs:complexType>
    </xs:element>
   </xs:complexType>
  </xs:element>
  <xs:attribute name="OID" type ="xs:ID" use="required"/>
  <xs:element name="Name" type="original-definition"/>
  <xs:element name="SpatialPoint"/>
 </xs:complexType>
</xs:element>
<xs:element name="SpatialPoint" minOccurs="0" maxOccurs ="unbounded">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="No." type="xs:integer"/>
   <xs:element ref="point"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
<xs:element name="point" minOccurs="1" maxOccurs="unbounded">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="Xaxis" type="xs:integer"/>
   <xs:element name="Yaxis" type="xs:integer"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
```

3. The subclass *DarkCloud* is mapped as follow:

```
<xs:element name="DarkCloud" minOccurs="0" maxOccurs="unbounded">
 <xs:complexType>
  <xs:element name="fuzzyGeneration" >
   <xs:complexType>
    <xs:extension base="Cloud"/>
    <xs:element name="Val">
     <xs:complexType>
      <xs:attribute name ="Poss" type ="xsd:fuzzy" default="1.0"/>
     </xs:complexType>
    </xs:element>
   </xs:complexType>
  </xs:element>
  <xs:element name="Rain"/>
  <xs:element name="TimeInterval"/>
  <xs:element name="Direction_1"/>
 </xs:complexType>
</xs:element>
<xs:element name="Rain" minOccurs="0" maxOccurs="unbounded">
 <xs:complexType>
  <xs:element name="Val"/>
 </xs:complexType>
</xs:element>
<xs:element name="Val" minOccurs="0" maxOccurs="unbounded">
 <xs:complexType>
  <xs:attribute name="Poss" type="xsd:fuzzy" default="1.0"/>
 </xs:complexType>
</xs:element>
<xs:element name="TimeInterval">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="timeStart"/>
   <xs:element name="timeEnd"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
<xs:element name="timeStart" minOccurs="0" maxOccurs="unbounded">
 <xs:simpleType>
  <xs:restriction base="xs:dateTime">
  <xs:minInclusive value="valueStart"/>
  <xs:maxInclusive value="valueEnd"/>
  </xs:restriction>
 </xs:simpleType>
</xs:element>
<xs:element name="timeEnd" minOccurs="0" maxOccurs="unbounded">
 <xs:simpleType>
  <xs:restriction base="xs:dateTime">
  <xs:minInclusive value="valueStart"/>
  <xs:maxInclusive value="valueEnd"/>
  </xs:restriction>
 </xs:simpleType>
</xs:element>
<xs:element name="Direction_1" minOccurs="1" maxOccurs="unbounded">
 <xs:complexType>
  <xs:element name="TimePoint"/>
  <xs:sequence>
   <xs:element ref="degree"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
<xs:element name="degree" minOccurs="2" maxOccurs="2">
 <xs:simpleType>
  <xs:restriction base="xs:integer">
  <xs:minInclusive value="degreeStart"/>
  <xs:maxInclusive value="degreeEnd"/>
  </xs:restriction>
 </xs:simpleType >
</xs:element>
<xs:element name="TimePoint" minOccurs="0" maxOccurs="unbounded">
 <xs:restriction base="xs:dateTime">
 <xs:minInclusive value="valueStart"/>
 <xs:maxInclusive value="valueEnd"/>
 </xs:restriction>
</xs:element>
```

4. The subclass *WhiteCloud* is mapped as follow:

```
<xs:element name="WhiteCloud" minOccurs="0" maxOccurs ="unbounded">
 <xs:complexType>
  <xs:element name="fuzzyGeneration" >
   <xs:complexType>
    <xs:extension base="Cloud"/>
    <xs:element name="Val">
     <xs:complexType>
      <xs:attribute name ="Poss" type ="xs:fuzzy" default="1.0"/>
     </xs:complexType>
    </xs:element>
   </xs:complexType>
  </xs:element>
  <xs:element name="Sunshade" type="original-definition"/>
  <xs:element name="Direction_2"/>
 </xs:complexType>
</xs:element>
<xs:element name="Direction_2" minOccurs="0" maxOccurs="unbounded">
 <xs:complexType>
  <xs:element name="MotionValue">
     <xs:simpleType>
      <xs:restriction base="xs:integer">
      <xs:minInclusive value="valueStart"/>
      <xs:maxInclusive value="valueEnd"/>
      </xs:restriction>
     </xs:simpleType>
  </xs:element>
  <xs:sequence>
   <xs:element name="degree" type="xs:integer"/>
   <xs:element name ="val"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
<xs:element name="Val" minOccurs="0" maxOccurs="unbounded">
 <xs:complexType>
  <xs:attribute name="Poss" type="xs:fuzzy" default="1.0"/>
 </xs:complexType>
</xs:element>
```

5. The linked and part-class *LivingQuarter* is mapped as follow:

```
<xs:element name="LivingQuarter" minOccurs="0" maxOccurs ="n">
 <xs:complexType>
  <xs:element name="fuzzyAssociation">
   <xs:complexType>
    <xs:extension base="City"/>
    <xs:element name="Val">
     <xs:complexType>
      <xs:attribute name ="Poss" type ="xs:fuzzy" default="1.0"/>
     </xs:complexType>
    </xs:element>
   </xs:complexType>
  </xs:element>
  <xs:element name="Year"/>
  <xs:element name="Market" type="original-definition"/>
  <xs:element name="SpatialLine"/>
 </xs:complexType>
</xs:element>
<xs:element name="Year" minOccurs="0" maxOccurs ="unbounded">
 <xs:complexType>
  <xs:element name ="Dist"/>
 </xs:complexType>
</xs:element>
<xs:complexType name="DistType" minOccurs="0" maxOccurs="unbounded">
 <xs:sequence>
  <xs:element name ="Val"/>
 </xs:sequence>
</xs:complexType>
<xs:element name="Val" minOccurs="0" maxOccurs="unbounded">
 <xs:complexType>
  <xs:element name="Vi" type="original-definition"/>
 </xs:complexType>
</xs:element>
<xs:element name="Spatialline">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="No." type="xs:integer"/>
   <xs:element ref="point "/>
  </xs:sequence>
  <xs:attribute name="offset" type="xs:integer"/>
 </xs:complexType>
</xs:element>
<xs:element name="point" minOccurs="2" maxOccurs="2">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="Xaxis" type="xs:integer"/>
   <xs:element name="Yaxis" type="xs:integer"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
```

6. The part-class *IndustrialDistrict* is mapped as follow:

```
<xs:element name="IndustrialDistrict" minOccurs="0" maxOccurs ="unbounded">
 <xs:complexType>
  <xs:element name="Produce" type="original-definition"/>
  <xs:element name="Factory" type="original-definition"/>
  <xs:element name="SpatialLine"/>
 </xs:complexType>
</xs:element>
<xs:element name="Spatialline">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="No." type="xs:integer"/>
   <xs:element ref="point "/>
  </xs:sequence>
  <xs:attribute name="offset" type="xs:integer"/>
 </xs:complexType>
</xs:element>
<xs:element name="point" minOccurs="2" maxOccurs="2">
 <xs:complexType>
  <xs:sequence>
   <xs:element name="Xaxis" type="xs:integer"/>
   <xs:element name="Yaxis" type="xs:integer"/>
  </xs:sequence>
 </xs:complexType>
</xs:element>
```

7. The whole-class *Land* is mapped as follow:

```
<xs:element name="Land" minOccurs="0" maxOccurs ="unbounded">
 <xs:complexType>
  <xs:element name ="fuzzyAggregation" >
   <xs:complexType>
    <xs:element ref = "LivingQuarter"/>
    <xs:element ref = "IndustrialDistrict"/>
    <xs:element name="Val" minOccurs ="0" maxOccurs ="unbounded">
     <xs:complexType>
      <xs:attribute name ="Poss" type ="xs:fuzzy" default="1.0"/>
     </xs:complexType>
    </xs:element>
   </xs:complexType>
  </xs:element>
 </xs:complexType>
</xs:element>
 <xs:element name="Population" type="original-definition"/>
 <xs:element name="TimePoint"/>
 </xs:complexType>
</xs:element>
<xs:element name="TimePoint" minOccurs="0" maxOccurs="unbounded">
 <xs:restriction base="xs:dateTime">
 <xs:minInclusive value="valueStart"/>
 <xs:maxInclusive value="valueEnd"/>
 </xs:restriction>
</xs:element>
```

FIGURE 9. The final transforming result.

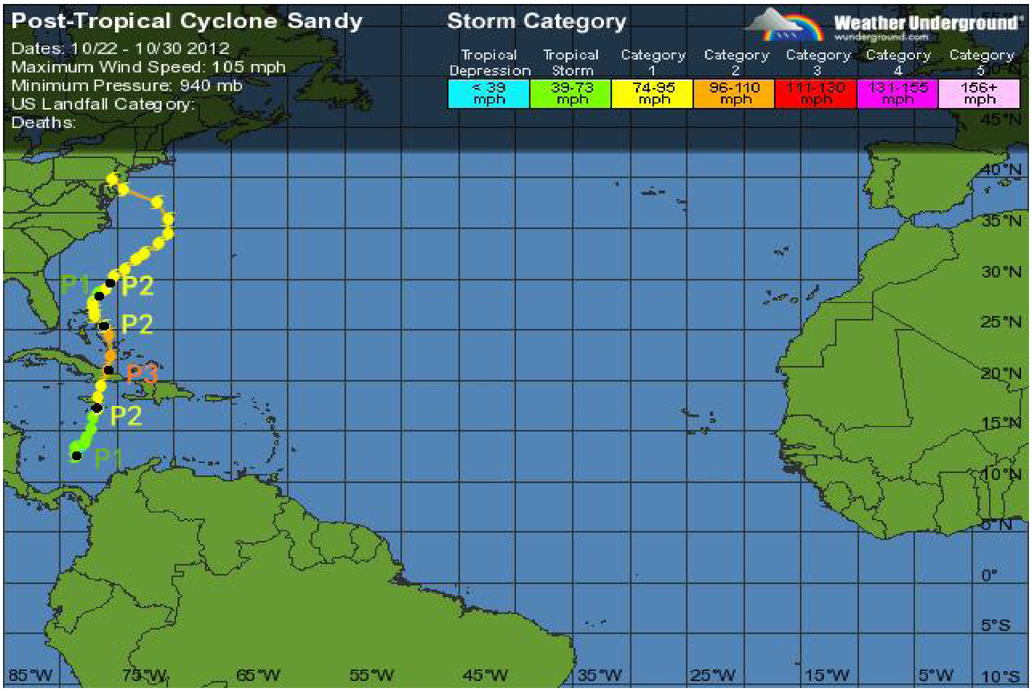


FIGURE 10. Track of post-tropical cyclone sandy.

IV. EXAMPLE

To understand the steps of Algorithm 1 and Algorithm 2, we present more details with the following supporting example in Fig. 8. In the example, there are fuzzy relationships between objects, deterministic general attributes, fuzzy general attributes, fuzzy spatial attributes and fuzzy temporal attributes. Through this example, we will use all of the rules that are introduced.

Let us analyze the fuzzy *OOSDM* database schema in Fig. 8. Firstly we use Rule 1 to create root element. Secondly, we use Rule 2-15 to map the whole class in the example. As to class Cloud, we firstly use Rule 2 to map class, and then we use Rule 3 to map *OID*. Attribute *name* is a general deterministic attribute, thus Rule 4 is used to map it. Fuzzy spatial point is mapped by Rule 8. As to class *DarkCloud*, Rule 15 is adopted to transform fuzzy time interval. Method *Rain* is fuzzy method and we apply the Rule 5 to transform it. We use Rule 14 and Rule 12 to transform fuzzy time point and fuzzy motion *direction_2* respectively.

We know that class *City is* a fuzzy class. Therefore, we firstly use Rule 2 and 6.Attribute fuzzy spatial region is transformed by Rule 10. For class *LivingQuarter* fuzzy attribute Year is mapped by applying Rule 7. We use Rule 9 to transform fuzzy *SpatialLine*. Class Land and *IndustrialDistrict* can be successfully transformed by using our approach. Lastly, we will map the fuzzy relationships. The class *DarkCloud* and *WhiteCloud* are two subclasses, and their superclass is Cloud. The degree of membership between *DarkCloud* and Cloud is α, and the degree of membership between *WhiteCloud* and Cloud is β. The Class *LivingQuarter* and *IndustrialDistrict* are two part-classes, and their whole-class is *Land*. The degree of membership between *LivingQuarter* and Land, between *IndustrialDistrict* and *Land* is ϕ, λ respectively. The relationship between class *Cloud* and class *City* is fuzzy association, the degree of membership is γ, and the involvement is *1:m*. The relationship between class *LivingQuarter* and class *City* is also fuzzy association, the degree of membership is η, and the involvement is *1:n*. Rule 16-18 are applied to map the relationships among classes. The final transforming result is shown in the following Fig. 9.

V. APPLICATIONS

In order to show validation of our approach, we apply it to meteorological events and show how to use our model record and transform fuzzy spatiotemporal data. Fig. 10 shows track of Post-Tropical Cyclone Sandy [24] from October 22, 2010 to October 30, 2010. In the following, we will analyze and model the Sandy and accomplish the transformation from *OOSDM* to XML.

A. ANALYZING AND MODELING POST-TROPICAL CYCLONE SANDY

In order to model the post-tropical cyclone Sandy accurately and efficiently, we will analyze the statistical data of Sandy used for the experiments.

In Fig. 10, several black points which are *P1*, *P2*, and *P3* are labeled. They represent different stages of the Sandy. *P1* represents Tropical Storm, *P2* represents Category 1 Hurricane and *P3* represents Category 2 Hurricane. The wind speed, and pressure in different stages are different. The values of wind speed, and pressure in one stage are in

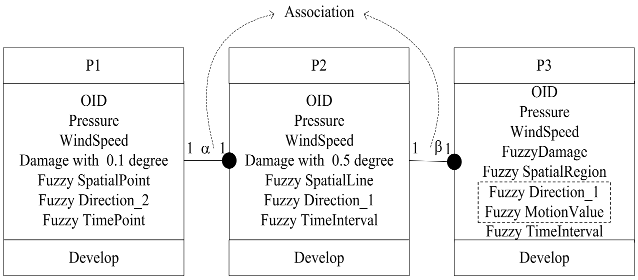


FIGURE 11. **The OOSDM schema of post-tropical cyclone sandy.**

a range. Table 2 shows the range of wind speed in *P1*, *P2* and *P3*. Table 3 shows the range of pressure in *P1*, *P2* and *P3*. According to our analysis, the value of wind speed is regular and the value of pressure is irregular. Thus wind speed will be an important factor in our model. For our model, there are different attributes in different stages, thus *P1*, *P2* and *P3* are three classes, but they have the relationship among classes.

As to *P1*, it is the staring stage. In the staring stage, we consider *P1* as *OID* since it is named by nature of hurricane and it is unique. *AT* of the spatiotemporal data can be multiple, such as wind speed and pressure. Meanwhile, the value of wind speed is in a regular range. For the *FAT*, the damage of wind is uncertain, and the possibility is low, it owns the first level of fuzziness. For *FST*, due to the speed and aggregation forces of wind are weak, so the focus of wind is not deterministic. We can use fuzzy spatial point and the second fuzzy motion direction to represent the state. There are *Lat* and *Lon* which can be used to represent fuzzy spatial point. For *FTM*, the staring time of hurricane is not precise, thus the fuzzy time point is used to model.

As to *P2*, it is the developing stage. In the developing stage, we consider *P2* as *OID* and it is unique. For *AT*, wind speed and pressure of hurricane are major attributes. For the *FAT*, we need pay close attention to the damage of wind. Consequently, we also use the damage of wind as a part of model, and the possibility increases. It belongs to the first level of fuzziness. For *FST*, the speed and aggregation forces of wind become strong, the influent scope of hurricane become large and the direction of hurricane is in an angle. We can use fuzzy spatial line and the first fuzzy motion direction to represent the state. The *Lat* and *Lon* can be used to represent fuzzy spatial line. For *FTM*, the developing time of hurricane is not precise, but we know developing time is behind the starting time and before the forming time, thus the fuzzy time interval is used to model.

As to *P3*, it is the forming stage. In the forming stage, we consider *P2* as *OID* and it is unique. For *AT*, we choose wind speed and pressure of hurricane. For the *FAT*, we also use the damage of wind as a part of model, but destructive force is fuzzy. So it is the third level of fuzziness. For *FST*, the speed and aggregation forces of wind become stronger, and the influent scope of hurricane become larger. We can use fuzzy spatial region and the first fuzzy motion direction to represent the state. At the same time, we should pay attention

FIGURE 12. **The XML schema of *P1*.**

to motion value. For *FTM*, the applying fuzzy time interval is suitable.

As to the relationship among *P1*, *P2*, and *P3*, it is a process of gradual development. So in the model, a method *Develop* should be added. The relationship among *P1*, *P2*, and *P3* is fuzzy association, and the degree of membership is high. According to the analysis, the *OOSDM* Schema is depicted in Fig. 11.

B. TRANSFORMING THE OOSDM SCHEMA OF POST-TROPICAL CYCLONE SANDY TO XML SCHEMA

In this subsection, we will accomplish the transformation on post-tropical cyclone Sandy from *OOSDM* to XML.

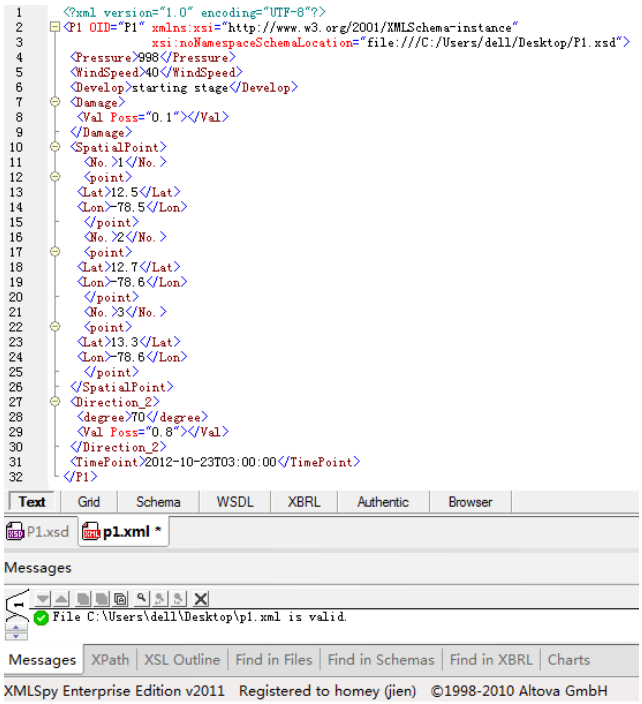


FIGURE 13. The XML document of *P1*.

There are three classes in the model, and every class contains *OID*, general attributes (or method), fuzzy general attributes, fuzzy spatial position, fuzzy spatial motion and fuzzy temporal attributes. There exists relationship among classes. As to the model above, the final transformation is depicted as follows.

According to our analysis and model, the class *P1* owns many attributes (or methods), such as *OID*, general attributes which contains pressure, wind speed whose value is from 39mph to 73mph, damage which is first level fuzziness, fuzzy spatial point attribute, fuzzy motion direction_2 attribute, fuzzy time point attribute and develop method. The class *P1* is mapped in Fig. 12.

From Fig. 12, we find that the XML Schema of *P1* is valid. Rule 1-18 are represented in formal language, as to real-word data, we should modify the XML Schema based on actual application. For example, the general attribute pressure whose data type changes from original-definition into integer, the general method develop whose data type changes form original-definition into string. Similarly, the type of *OID* is string. In the fuzzy spatial point, *Xaxis* is *Lat*, and *Yaxis* is *Lon*. The type of *Lat* and *Lon* is float. Post-Tropical Cyclone Sandy is during from October 22, 2010 to October 30, 2010, thus fuzzy time point is during from 2012-10-22T00:00:00 to 2012-10-31T23:59:59. Finally, the generating XML document of *P1* is depicted in Fig. 13.

VI. CONCLUSION

In this paper, a methodology of transforming fuzzy spatiotemporal data from object-oriented database to XML is proposed. At first, we introduce fuzzy spatiotemporal data, and then establish fuzzy object-oriented spatiotemporal data model and fuzzy XML spatiotemporal data model. In addition, we produce a set of rules to accomplish the transforming process and algorithms to deal with the exchange of fuzzy spatiotemporal data from object-oriented database to XML. Last, a real world example is taken to evaluate our methodology.

Future work will center on three sides. Firstly, we will consider more complex fuzzy spatiotemporal objects. Secondly, the automatic transformation without human intervention is another future work. Thirdly, we will further transform fuzzy spatiotemporal data from XML to object-oriented database.

REFERENCES

[1] S. Benferhat, J. Ben-Naim, O. Papini, and E. Würbel, "An answer set programming encoding of prioritized removed sets revision: Application to GIS," *Appl. Intell.*, vol. 32, no. 1, pp. 60–87, 2010.

[2] M. Dewar, K. Scerri, and V. Kadirkamanathan, "Data-driven spatio-temporal modeling using the integro-difference equation," *IEEE Trans. Signal Process.*, vol. 57, no. 1, pp. 83–91, Jan. 2009.

[3] P. Xu, Y. Sugano, and A. Bulling, "Spatio-temporal modeling and prediction of visual attention in graphical user interfaces," in *Proc. 34th Annu. ACM Conf. Human Factors Comput. Syst.*, New York, NY, USA, 2016, pp. 3299–3310.

[4] N. Pelekis, B. Theodoulidis, I. Kopanakis, and Y. Theodoridis, "Literature review of spatio-temporal database models," *Knowl. Eng. Rev.*, vol. 19, no. 3, pp. 235–274, Sep. 2004.

[5] A. Yazici and R. George, *Fuzzy Database Modeling*. Vienna, Austria: Physica-Verlag, 2013.

[6] A. Sözer, A. Yazıcı, H. Oğuztüzün, and O. Taş, "Modeling and querying fuzzy spatiotemporal databases," *Inf. Sci.*, vol. 178, no. 19, pp. 3665–3682, 2008.

[7] A. Frihida, D. J. Marceau, and M. Thériault, "Spatio-temporal object-oriented data model for disaggregate travel behavior," *Trans. GIS*, vol. 6, no. 3, pp. 277–294, Jun. 2002.

[8] F. Berzal, N. Marín, O. Pons, and M. A. Vila, "Managing fuzziness on conventional object-oriented platforms," *Int. J. Intell. Syst.*, vol. 22, no. 7, pp. 781–803, Jul. 2007.

[9] Z. M. Ma, W. J. Zhang, and W. Y. Ma, "Extending object-oriented databases for fuzzy information modeling," *Inf. Syst.*, vol. 29, no. 5, pp. 421–435, Jul. 2004.

[10] J. B. Joshi, W. G. Aref, A. Ghafoor, and E. H. Spafford, "Security models for Web-based applications," *Commun. ACM*, vol. 44, no. 2, pp. 38–44, Feb. 2001.

[11] A. Gaurav and R. Alhajj, "Incorporating fuzziness in XML and mapping fuzzy relational data into fuzzy XML," in *Proc. ACM Symp. Appl. Comput.*, New York, NY, USA, 2006, pp. 456–460.

[12] F. Rizzolo and A. A. Vaisman, "Temporal XML: Modeling, indexing, and query processing," *VLDB J.*, vol. 17, no. 5, pp. 1179–1212, Aug. 2008.

[13] B. Oliboni and G. Pozzani, "Representing fuzzy information by using XML schema," in *Proc. DEXA*, Turin, Italy, Sep. 2008, pp. 683–687.

[14] K. Turowski and U. Weng, "Representing and processing fuzzy information—An XML-based approach," *Knowl.-Based Syst.*, vol. 15, nos. 1–2, pp. 67–75, Jan. 2002.

[15] M. Murata, D. Lee, M. Mani, and K. Kawaguchi, "Taxonomy of XML schema languages using formal language theory," *ACM Trans. Internet Technol.*, vol. 5, no. 4, pp. 660–704, Nov. 2005.

[16] L. Yan, Z. M. Ma, and J. Liu, "Fuzzy data modeling based on XML schema," in *Proc. ACM Symp. Appl. Comput.*, New York, NY, USA, 2009, pp. 1563–1567.

[17] X. Wu, C. Zhang, and S. Zhang, "Database classification for multi-database mining," *Inf. Syst.*, vol. 30, no. 1, pp. 71–88, Mar. 2005.

[18] T. Naser, R. Alhajj, and M. J. Ridley, "Reengineering XML into object-oriented database," in *Proc. IEEE Int. Conf. Inf. Reuse Integr.*, Las Vegas, NV, USA, Jul. 2008, pp. 1–6.

[19] T. Naser, R. Alhajj, and M. J. Ridley, "Two-way mapping between object-oriented databases and XML," *Informatica*, vol. 33, no. 3, pp. 297–308, 2009.

[20] T. Naser, K. Kianmehr, R. Alhajj, and M. J. Ridley, "Transforming object-oriented databases into XML," in *Proc. IEEE Int. Conf. Inf. Reuse Integr.*, Las Vegas, IL, USA, Aug. 2007, pp. 600–605.

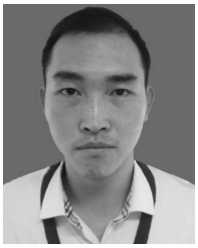
[21] J. F. Terwilliger, P. A. Bernstein, and S. Melnik, "Full-fidelity flexible object-oriented XML access," *Proc. VLDB Endowment*, vol. 2, no. 1, pp. 1030–1041, Aug. 2009.

[22] J. Liu, Z. M. Ma, and X. Feng, "Formal approach for reengineering fuzzy XML in fuzzy object-oriented databases," *Appl. Intell.*, vol. 38, no. 4, pp. 541–552, Jun. 2013.

[23] J. Liu and Z. M. Ma, "Formal transformation from fuzzy object-oriented databases to fuzzy XML," *Appl. Intell.*, vol. 39, no. 3, pp. 630–641, Oct. 2013.

[24] *Weather Underground*. Accessed: 2016. [Online]. Available: https://www.wunderground.com

LUYI BAI received the Ph.D. degree from Northeastern University, China. He is currently an Associate Professor with Northeastern University, Qinhuangdao, China. He has authored or co-authored papers in several journals. His current research interests include uncertain databases and fuzzy spatiotemporal XML data management. He is a member of CCF.



ZHIYI JIA was born in Anyang, Henan, China, in 1990. He is currently pursuing the master's degree in computer system architecture with Northeastern University, China. His current research interests include fuzzy spatiotemporal XML data management and uncertain object-oriented database.

JIEMIN LIU received the Ph.D. degree from Northeastern University, China. He is currently a Professor and the Dean of Northeastern University, Qinhuangdao, China. He has authored or co-authored several papers and academic monographs. His current research interests include next-generation network and information management system. He is also the Vice Chairman of ACM China Council, Qinhuangdao, and the Chairman of CCF, Qinhuangdao.