Available online at www.sciencedirect.com
ScienceDirect
Fuzzy Sets and Systems 186 (2012) 26–46
FUZZY sets and systems
www.elsevier.com/locate/fss

Modeling fuzzy information in UML class diagrams and object-oriented database models

Z.M. Ma[a,*], Li Yan[b], Fu Zhang[a]

[a] College of Information Science and Engineering, Northeastern University, Shenyang, Liaoning 110819, China
[b] School of Software, Northeastern University, Shenyang, Liaoning 110819, China

Received 26 July 2009; received in revised form 26 June 2011; accepted 27 June 2011
Available online 5 July 2011

Abstract

Conceptual data modeling has become essential for non-traditional application areas. Some conceptual data models have been proposed as tools for database design and object-oriented database modeling. Information in real-world applications is often vague or ambiguous. Currently, a little research is underway on modeling the imprecision and uncertainty in conceptual data modeling and the conceptual design of fuzzy databases. The unified modeling language (UML) is a set of object-oriented modeling notations and a standard of the object management group (OMG) with applications to many areas of software engineering and knowledge engineering, increasingly including data modeling. This paper introduces different levels of fuzziness into the class of UML and presents the corresponding graphical representations, with the result that UML class diagrams may model fuzzy information. The fuzzy UML data model is also formally mapped into the fuzzy object-oriented database model.

Keywords: Database modeling; Fuzzy information; UML class diagrams; Object-oriented database model

1. Introduction

One of the major areas of database research has been the continuous effort to enrich existing database models with a more extensive collection of semantic concepts. Database models have been developed from hierarchical and network database models to the relational database model. As computer technology moves into non-traditional applications such as CAD/CAM, knowledge-based systems, and multimedia and Internet systems, many software engineers feel the limitations of relational databases in these data- and knowledge-intensive applications. Therefore, some non-traditional data models have been proposed for databases, such as the entity-relationship (ER) model [1], the enhanced (or extended) entity-relationship (EER) model (e.g., [2]), the object-oriented (OO) database model, the object-relational database model and the logic database model. Among these database models, the object-relational database model combines the robustness of the relational database model with the powerful modeling capabilities of the object-oriented paradigm [3].

* Corresponding author.
E-mail address: mazongmin@ise.neu.edu.cn (Z.M. Ma).

0165-0114/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.fss.2011.06.015

The unified modeling language (UML) [4,5] is a general-purpose modeling language used in the field of object-oriented software engineering and has been standardized by the Object Management Group (OMG). UML includes elements such as activities, actors, business processes, database schemas, (logical) components, programming language statements and reusable software components, combining techniques from data modeling (entity-relationship diagrams), business modeling (work flows), object modeling and component modeling. UML can be used with all processes throughout the software development life cycle and across different implementation technologies. The power of UML can be applied to many areas of software engineering and knowledge engineering [6]. For example, UML can describe the complete development of relational and object-relational databases for business requirements [7]. UML is being applied to database modeling [8–10], and UML has been used to conceptual model XML [11] and OWL in the Semantic Web [12]. Note that the use of UML for database modeling does not imply that UML is perfectly suited for use as a database modeling framework or that it is the only choice to accomplish this task. In practice, data modeling is conducted with a variety of design methods within different organizations, depending on the skills of the involved human resources.

While UML and object-oriented database models provide powerful object-oriented modeling capabilities, they suffer from some inadequacy of necessary semantics. One of these inadequacies can be generalized as the inability to handle imprecise and uncertain information [13]. In real-world applications, information is often imperfect. One of the semantic needs not adequately addressed by the traditional data models is that of uncertainty. Traditional data models assume that the models are a correct reflection of the world and further assume that the stored data is known, accurate and complete. It is rarely the case in real life that all or most of these assumptions are met. Fuzzy sets [14] and possibility theory [15] are embedded in many of the existing approaches to dealing with imperfect information.

Fuzzy information has been extensively investigated in the context of the relational database model [16–22]. However, the classical relational database model and its fuzzy extension do not satisfy the need to model complex objects with inherent imprecision and uncertainty. The requirements of modeling complex objects and information imprecision and uncertainty can be found in many application domains and have challenged the current database technology [23,24]. Because the object-oriented database model can represent complex object structures without fragmenting the aggregate data and can also depict complex relationships among attributes, current efforts have concentrated on the fuzzy object-oriented databases and some related notions such as class, superclass/subclass, and inheritance [25–34]. More recently, fuzzy object-relational databases have been proposed [35] that combine the characteristics of fuzzy relational databases and fuzzy object-oriented databases. Note that almost no research is currently underway on modeling fuzzy information in the conceptual data models, with most research instead focusing on modeling uncertainty at the data level; less knowledge exists about uncertainty at the conceptual model level. This is particularly true in design methodologies used to design fuzzy database schemas.

Zvieli and Chen [36] applied fuzzy set theory to some of the basic ER concepts and introduced fuzzy entity sets, fuzzy-relationship sets and fuzzy attribute sets (the first level of fuzziness) as well as fuzziness in entity and relationship occurrences (the second level of fuzziness) and in attribute values (the third level of fuzziness). Based on the fuzzy ER model, a methodology for the design and development of fuzzy-relational databases is proposed in Chaudhry et al. [37] through the rules developed for mapping the fuzzy ER schema to the fuzzy-relational database schemas. Other efforts to extend the ER model can be found in Ruspini [38], Vandenberghe [39], Vert et al. [40]. Without including graphical representations, the fuzzy extensions of several major EER concepts (including superclass/subclass, generalization/specialization, category and the subclass with multiple superclasses) are introduced in Chen and Kerre [41]. A full-fledged fuzzy extension to the EER model and the corresponding graphical representations are presented in Ma et al. [42], along with the formal approach to mapping a fuzzy EER model to a fuzzy object-oriented database schema. In Galindo et al. [43], the fuzzy EER models are extended by relaxing some constraints with fuzzy quantifiers. In addition to the ER/EER model, the IFO data model, a graph-based conceptual data model proposed in Abiteboul and Hull [44], is also extended to deal with fuzzy information. In Vila et al. [45], several types of imprecision and uncertainty are incorporated into the attribute domain of the object-based data model, such as the values without semantic representation, the values with semantic representation and disjunctive meaning, the values with semantic representation and conjunctive meaning, and the representation of uncertain information. However, some major concepts in object-based modeling (e.g., superclass/subclass and class inheritance) are not discussed. In addition to the attribute-level uncertainty, Vila et al. [45] consider the uncertainty to be at the level of object and class. In Yazici et al. [46], two levels of uncertainty are considered based on similarity relations, namely the level of attribute values and the level of entity instances, giving rise to the ExIFO model. Also, the mapping from the ExIFO model into the fuzzy nested

relational database schemas is described. The mapping from the IF_2O model, an extended fuzzy IFO model based on fuzzy set and possibility theory, into the fuzzy-relational database schemas is described in Ma [47]. A fuzzy semantic model (FSM) that draws on some constructs found in several fuzzy conceptual data models (e.g., fuzzy ER/EER and IFO data models), is proposed in Bouaziz et al. [48], and a query language adapted to FSM-based databases is introduced. Although some studies perform fuzzy conceptual modeling, only a few incorporate fuzziness with the UML model. Sicilia and Mastorakis [13] define several new fuzzy constructs for the extended UML model. In addition, UML is extended with some fuzzy constructs for multimedia database applications [49,50] and spatiotemporal database applications [51]. The proposed model supports the representation of uncertainty at the attribute, object/class and class/subclass levels.

Conceptual data models can capture and represent rich and complex semantics at a highly abstract level. The design of large and complex databases in applications generally starts with designing the conceptual data models, which are then mapped into the logical database models. Various conceptual data models have been used for the conceptual design of database schemas. For example, relational databases are designed by first developing a high-level conceptual data model, such as an ER model, and then an ER-diagram is mapped to a database schema [52]. In information management of a multimedia database, for example, the attributes of an image such as color, the description of an object and the spatial relations between objects may be imprecise [23,24], such as "*bright*" in color and "*near*" in spatial relation, which are highly useful in intelligent semantics-based image retrieval systems. To represent and handle complex objects with imprecise and uncertain information, a challenge in many data- and knowledge-intensive application areas, and in particular to provide a conceptual design methodology for the fuzzy object-oriented databases, this paper models fuzzy information via the UML data model and the object-oriented database model. Fuzzy sets and possibility distributions introduce different levels of fuzziness into the class of the UML model, enabling the UML class diagrams to model fuzzy information. Based on the corresponding graphical representations of the fuzzy UML data model, this paper describes the formal mapping of the fuzzy UML data model into the fuzzy object-oriented database model.

This paper represents a departure from the existing fuzzy extensions to the EER and IFO models, which only consider the second level of fuzziness. First, we introduce several major concepts into the UML class model such as association and dependency. Second, these concepts allow for the full consideration of the first and second levels of fuzziness in classes. Finally, the fuzzy UML data model is mapped into the fuzzy object-oriented database model. The contribution of this paper is the full development of an OO conceptual modeling methodology for the modeling of fuzzy information.

The remainder of this paper is organized as follows. Section 2 provides basic knowledge concerning fuzzy sets and fuzziness in conceptual data models. The fuzzy extension to the UML class model is developed in Section 3. Section 4 presents the formal mapping from the fuzzy UML class model to the fuzzy object-oriented database model, and Section 5 concludes this paper.

2. Fuzzy sets and fuzziness in conceptual data models

Different models have been proposed to handle different categories of data quality (or the lack thereof). Five basic kinds of imperfection have been identified [53], which are *inconsistency*, *imprecision*, *vagueness*, *uncertainty*, and *ambiguity*. Rather than providing the formal definitions of these types of imperfect information, we explain their meanings here.

Inconsistency is a type of semantic conflict, meaning that the same aspect of the real world is irreconcilably represented more than once in a database or in several different databases. For example, the *year of birth* of *George* is simultaneously stored as 1966 and 1967. Information inconsistency usually arises from information integration.

Intuitively, imprecision and vagueness are relevant to the content of an attribute value, which means that a choice must be made from a given range (interval or set) of values without knowing which one should be chosen. In general, vague information is represented by linguistic values. For example, assume that we do not know the exact year of birth of two persons named *Michael* and *John*, but we do know that the *year of birth* of *Michael* may be 1958, 1959, 1960, or 1961, and *John* is a baby. The information about *Michael*'s year of birth is imprecise, denoted by a set of values {1958, 1959, 1960, 1961}. The information about *John*'s year of birth is vague one, and can be denoted by the linguistic value, "*infant*".

Uncertainty indicates that we can apportion some, but not all, of our belief to a given value or group of values. For example, the possibility that the *year of birth* of *Chris* is 1955 right now should be 98%. This paper does not consider

random uncertainty, which can be described using probability theory. The ambiguity means that some elements of the model lack complete semantics, leading to several possible interpretations.

Generally, several different kinds of imperfection can co-exist with respect to the same piece of information. For example, the *year of birth* of *Michael* is a set of values {1958, 1959, 1960, 1961} with possibilities of 70%, 95%, 98%, and 85%, respectively. Imprecision, uncertainty, and vagueness are the three major types of imperfect information and can be modeled with fuzzy sets [14] and possibility theory [15]. Many current approaches to imprecision and uncertainty are based on the theory of fuzzy sets [54,55].

Let U be a universe of discourse and F be a fuzzy set in U. A membership function:

$$\mu_F : U \to [0, 1]$$

is defined for F, where $\mu_F(u)$ for each $u \in U$ denotes the membership degree of u in the fuzzy set F. Thus, the fuzzy set F is described as follows:

$$F = \{(u_1, \mu_F(u_1)), (u_2, \mu_F(u_2)), \ldots, (u_n, \mu_F(u_n))\}.$$

When the membership degree $\mu_F(u)$ above is explained as a measure of the possibility that a variable X has the value u, where X takes on values in U, a fuzzy value is described by the possibility distribution π_X [15].

$$\pi_X = \{(u_1, \pi_X(u_1)), (u_2, \pi_X(u_2)), \ldots, (u_n, \pi_X(u_n))\}.$$

Here, $\pi_X(u_i),\ u_i \in U$ denotes the possibility that u_i is true. Let π_X be the representation of the possibility distribution for the fuzzy value of a variable X. This means that the value of X is fuzzy, and X may take on one of the possible values $u_1, u_2, \ldots,$ and u_n, with each possible value (say u_i) associated with a possibility degree (say $\pi_X(u_i)$).

Fuzzy set theory was first applied to some of the basic ER concepts in Zvieli and Chen [36]. This work introduced the fuzzy entity type set, fuzzy-relationship type set and fuzzy attribute set of entity types (or relationship types) in addition to fuzziness in entity occurrences, relationship occurrences and attribute values, constituting the following three levels of fuzziness in the ER model.

- At the first level, the entity type set, relationship type set and attribute set of entity (or relationship) types may be fuzzy.
- The second level is related to the fuzzy occurrences of entities and relationships.
- The third level concerns the fuzzy values of attributes in entities and relationships.

An ER model generally includes some entity types, which constitute a set of entity types and some relationship types, which constitute a set of relationship types. In addition, an entity type or a relationship type generally includes some attributes, which constitute a set of attributes. In the fuzzy ER model, the first level of fuzziness means that these three kinds of sets may be fuzzy sets. Let the entity type set be fuzzy. Then, an entity type belongs to the entity type set with a membership degree in [0, 1] that indicates the possibility that this entity type belongs to the entity type set. Similarly, if the relationship type set is a fuzzy set, a relationship type belongs to the relationship type set with a membership degree in [0, 1] that indicates the possibility that this relationship type belongs to the relationship type set; if the attribute set of an entity type or a relationship type is a fuzzy set, then an attribute belongs to the attribute set with a membership degree in [0, 1] that indicates the possibility that this attribute belongs to the attribute set. For example, consider the membership values for entity types, relationship types and attributes. Suppose that we have an ER model about a library that includes the two entity types *Book* and *Book Store*, and that there is a relationship *PurchasedFrom* between these two entity types. The model assumes that *Book Store* is a fuzzy entity type with a membership grade of 0.6. Then, *PurchasedFrom* is a fuzzy-relationship type with a membership grade of 0.6. Also, *Book* may contain an attribute *Dimensions* in addition to the attributes *ID*, *Title*, *Authors*, *ISBN*, *Publisher*, etc., and *Dimensions* is a fuzzy attribute with a membership grade of 0.4.

3. UML modeling of fuzzy data

This section extends the UML class diagram to represent fuzzy information. Because the constructs of UML contain *class* and *relationships*, the extension to these constructs should be conducted based on fuzzy sets. For this purpose, we first give a formal description of the UML class diagram.

A UML class diagram is a tuple $D = (C, A, R, O, M, S)$, in which C is a finite set of classes, A is a finite set of attributes, R is a finite set of relationships, O is a finite set of objects, M is a finite set of methods, and S is a finite set of constraints. This paper focuses solely on the classes, attributes, relationships, and objects, yielding a simple UML class diagram model: $D = (C, A, R, O)$, where $C = \{c_1, c_2, \ldots, c_k\}$, $A = \{a_1, a_2, \ldots, a_l\}$, $R = \{r_1, r_2, \ldots, r_m\}$, and $O = \{o_1, o_2, \ldots, o_n\}$. Then, we have:

- $R \subseteq C \times C$ is a binary relation that represents the generalization, aggregation, association or dependency.
- For $c_i \in C(1 \leq i \leq k)$, $A(c_i)$ represents a set of attributes of c_i. Clearly $A(c_i) \subseteq \{a_1, a_2, \ldots, a_l\}$, i.e., $A(c_i) \subseteq A$. For $a_j \in A(1 \leq j \leq l)$, $a_j(c_i)$ denotes the attribute a_j of c_i. In the context of the given c_i, a_j is used instead of a_j (c_i).
- For $c_i \in C(1 \leq i \leq k)$, $O(c_i)$ means a set of objects that c_i contains. Here, $O(c_i) \subseteq \{o_1, o_2, \ldots, o_n\}$, i.e., $O(c_i) \subseteq O$. For $o_p \in O(1 \leq p \leq n)$ and $a_j \in A(1 \leq j \leq l)$, o_p (c_i) denotes the object o_p of c_i, and o_p $(a_j(c_i))$ denotes the value of object o_p on attribute a_j. In the context of the given c_i, o_p is used instead of o_p (c_i) and o_p (a_j) is used instead of $o_p(a_j(c_i))$.

To accommodate fuzzy information in the UML class diagram, the UML class diagram model must be extended by using the fuzzy set and fuzzy logic. Formally, a fuzzy UML class diagram is a tuple $\tilde{D} = (\tilde{C}, \tilde{A}, \tilde{R}, \tilde{O})$, where $\tilde{C}$ is a fuzzy set of classes, $\tilde{A}$ is a fuzzy set of attributes, $\tilde{R}$ is a fuzzy set of relationships, and $\tilde{O}$ is a fuzzy set of objects. In the following we investigate the details of the fuzzy UML class diagram.

3.1. Fuzzy class

Theoretically, a class can be considered from two different viewpoints:

(a) an extensional class, where the class is defined by the list of its object instances, and
(b) an intensional class, where the class is defined by a set of attributes and their admissible values.

A subclass defined from its superclass by means of the inheritance mechanism, and this can be regarded as the special case of (b) above.

Objects with the same properties are grouped into classes. Suppose that some fuzzy objects have similar properties and that a class is defined by these objects. These objects belong to the class with membership degrees of [0, 1], making it a fuzzy class. Also, for an intensional class, the domain of a class attribute may be fuzzy. As a result, some objects may have fuzzy values on this attribute, making the corresponding class a fuzzy class. Finally, a class is produced by a fuzzy class by means of specialization, or a class is produced by some classes (in which at least one class is fuzzy) by means of generalization. At this point, the produced class (subclass for the former and superclass for the later) is fuzzy.

Following Zvieli and Chen [36], the context of the class contains three levels of fuzziness in $\tilde{D} = (\tilde{C}, \tilde{A}, \tilde{R}, \tilde{O})$.

(a) The first level of fuzziness evaluates the extent to which the class belongs to the data model as well as the fuzziness of the content (in terms of attributes) of the class. At this point, we have a fuzzy set of classes $\tilde{C}$ and then a class (e.g., c_i) is the class of $\tilde{C}$ with a membership degree (e.g., $\mu_{\tilde{C}}(c_i)$); or for a class (e.g., c_i), we have a fuzzy set of its attributes (e.g., $A(c_i)$) and then an attribute (e.g., a_j (c_i)) is the attribute of $A(c_i)$ with a membership degree (e.g., $\mu_{\tilde{A}}(a_j(c_i))$).
(b) The second level of fuzziness evaluates the extent to which some objects belong to a class. An object is a fuzzy one if it contains at least one fuzzy attribute value. Then such an object (e.g., $o_p(c_i)$) is the object of class (e.g., $O(c_i)$) with a membership degree (e.g., $\mu_{\tilde{O}}(o_p(c_i))$).
(c) The third level of fuzziness is that of the attribute values of the objects of the class. An attribute in a class defines a value domain. When this domain is a fuzzy subset or a set of a fuzzy subset, the value of an object on the attribute, say $o_p(a_j(c_i))(1 \leq p \leq n)$, is a fuzzy one represented by a possibility distribution, say $\{(v_1, \pi(v_1)), (v_2, \pi(v_2)), \ldots, (v_q, \pi(v_q))\}$. Here, $\pi(v_s)(1 \leq s \leq q)$ denotes the possibility that $o_p(a_j(c_i))$ has value v_s.

Consider a class diagram model of the preliminary design of the product *Car*. Suppose that there is a class named *GPS* and it is unknown whether the model includes this class at the preliminary design stage. Class GPS possibly belongs to the model with a membership degree of 0.90. Then, GPS is a class with the first level of fuzziness. Also, at this point

it is possible that a variety of GPS products may be selected, say *CS*3*K*(*SONY*), *HDR-TG*5*V*(*SONY*), *NV-U*3*C*(*SONY*), *nüvi* 465*T*(*GARMIN*) and *GORILLA*(*SANYO*), and these objects belong to class GPS with membership degrees of 0.7, 0.9, 1.0, 1.0, and 0.8, respectively. Then, GPS is a class with the second level of fuzziness. Let class GPS contains an attribute named *quality* and let it be possible that the values of some objects are fuzzy on this attribute, with the value of nüvi 465T(GARMIN) on this attribute as *very good*. Then, GPS is a class with the third level of fuzziness.

The three levels of fuzziness in the class constitute the foundation of the fuzzy UML class diagram, so their soundness is crucial. First, consider the first level of fuzziness. For the fuzzy set $\tilde{C}$ of classes and any class $c_i(1 \leq i \leq k)$, the degree to which c_i belongs to $\tilde{C}$ is $\mu_{\tilde{C}}(c_i)(0 \leq \mu_{\tilde{C}}(c_i) \leq 1)$. This implies that for the traditional UML class diagram with no imprecise or uncertain information, either $\mu_{\tilde{C}}(c_i) = 0$, which means that c_i definitely does not belong to $\tilde{C}$, or $\mu_{\tilde{C}}(c_i) = 1$, which means that c_i must belong to $\tilde{C}$. At this point, $\tilde{C}$ becomes a crisp set of classes. Also, for the class $c_i(1 \leq i \leq k)$ and the attribute $a_j(c_i)(1 \leq j \leq l)$ of c_i, the degree to which a_j belongs to $A(c_i)$ is $\mu_{\tilde{A}}(a_j(c_i))(0 \leq \mu_{\tilde{A}}(a_j(c_i)) \leq 1)$. Then, either $\mu_{\tilde{A}}(a_j(c_i)) = 0$ or $\mu_{\tilde{A}}(a_j(c_i)) = 1$ under the traditional information environment (no imprecision or uncertainty at all). The former situation indicates that a_j is not an attribute of c_i, while the latter situation indicates that a_j must be an attribute of c_i. Second, let us focus on the second level of fuzziness. For the class $c_i(1 \leq i \leq k)$ and an object instance $o_p(c_i)(1 \leq p \leq n)$ of c_i, the degree to which o_p belongs to $O(c_i)$ is $\mu_{\tilde{O}}(o_p(c_i))$ $(0 \leq \mu_{\tilde{O}}(o_p(c_i)) \leq 1)$. Then, under the traditional information environment, either $\mu_{\tilde{O}}(o_p(c_i)) = 0$ or $\mu_{\tilde{O}}(o_p(c_i)) = 1$, which means that o_p is not the object of c_i or that o_p must be the object of c_i, respectively. Finally, consider the third level of fuzziness. The attribute value of object o_p $(a_j(c_i))$ is represented by a possibility distribution $\{\pi(v_1)/v_1, \pi(v_2)/v_2, \ldots, \pi(v_q)/v_q\}$. The possibility that $o_p(a_j(c_i))$ has the value $v_s(1 \leq s \leq q)$ is $\pi(v_s)(0 \leq \pi(v_s) \leq 1)$. Also, when there is no imprecise or uncertain information, either $\pi(v_s) = 0$ or $\pi(v_s) = 1$. As a result, $\{(v_1, \pi(v_1)), (v_2, \pi(v_2)), \ldots, (v_q, \pi(v_q))\}$ is reduced to a crisp set and $o_p(a_j(c_i))$ takes on a crisp attribute value. In summary, the traditional UML class diagram is simply a special case of the fuzzy UML class diagram. As it is an extension of the traditional UML class diagram, the fuzzy UML class diagram with three levels of fuzziness can be reduced to the traditional diagram in the absence of imprecise and uncertain information. Therefore, the extension of the three levels of fuzziness to classes is sound.

To model the first level of fuzziness, i.e., an attribute or a class with a membership degree, the attribute or class name should be followed by a pair of words WITH *mem* DEGREE, where $0 \leq mem \leq 1$, which is used to indicate the degree to which the attribute belongs to the class or to which the class belongs to the data model [29,32]. For example, "*Employee WITH 0.6 DEGREE*" and "Office Number *WITH* 0.8 *DEGREE*" are a class and an attribute, respectively, with the first level of fuzziness. Generally, an attribute or class will not be declared when its membership degree is 0. In addition, "WITH 1.0 DEGREE" can be omitted when the membership degree of an attribute or class is 1. Note that attribute values may be fuzzy. To model the third level of fuzziness, the keyword FUZZY is introduced and placed in front of the attribute name. For the second level of fuzziness, we must indicate the membership degree to which an instance of the class belongs to the class. For this purpose, an additional attribute is introduced into the class to represent the membership degree of the instance to the class, with the domain [0, 1]. We denote such a special attribute with μ. A class with the second level of fuzziness is denoted by a rectangle with a dashed outline.

Fig. 1 shows the fuzzy class *Young Employee*. Here, the attribute *Year of Birth* may take on fuzzy values; namely, its domain is fuzzy. It is unclear whether the class *Young Employee* has the attribute *Spouse*, but we do know that young employees have spouses with intermediate possibility, say 0.5. Therefore, the attribute *Spouse* uncertainly belongs to the class *Young Employee*. This class has fuzziness at the first level, and we use "with 0.5 membership degree" to describe the fuzziness in the class definition. In addition, we cannot determine whether an object is an instance of the class because the class is fuzzy. Therefore, an additional attribute μ is introduced into the class for this purpose.

3.2. Fuzzy generalization

Inheritance is a mechanism in object-oriented data modeling that allows a class referred to as a subclass to inherit attributes and methods from another class called its superclass. As a result, inheritance allows the definition of superclasses and subclasses, and classes are organized in inheritance hierarchies in which the definitions of attributes and methods are inherited among classes. Because a subclass is a specialization of its superclass, any one object belonging to the subclass must belong to the superclass. This characteristic can be used to determine whether two classes have a subclass/superclass relationship.

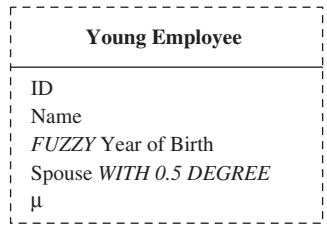

Fig. 1. A fuzzy class.

In the fuzzy UML data model, classes may be fuzzy. A class produced from a fuzzy class by means of inheritance may be fuzzy. If the former is still called a subclass and the latter is a superclass, the subclass/superclass relationship is fuzzy. In other words, a class is a subclass of another class with the membership degree [0, 1] at this moment. Also, the fact that a class may be a fuzzy class results in a (fuzzy) object belonging to this class with a membership degree. A threshold is needed to determine the subclass/superclass relationship using objects. We have developed the following method for determining the subclass/superclass relationship.

(a) For any (fuzzy) object, the membership degree to which it belongs to the subclass is greater than or equal to a given threshold, and
(b) the membership degree to which it belongs to the subclass is less than or equal to the membership degree to which it belongs to the superclass.

The subclass is then a subclass of the superclass with a membership degree, which is the minimum of the membership degrees to which these objects belong to the subclass. Here, the given threshold is used to set the confidence with which two classes have a subclass/superclass relationship with membership degree. Generally, two kinds of objects of the subclass can be identified: the objects that have membership degrees less than the given threshold and objects that have membership degrees greater than or equal to the given threshold. With the given threshold, two classes have a subclass/superclass relationship as long as the latter objects have membership degrees that are less than or equal to the membership degrees to which they belong to the superclass. The former objects are not applied to determine whether they have membership degrees less than or equal to the membership degrees to which they belong to the superclass. If the threshold is not set, then the two classes do not have a subclass/superclass relationship when an object of the subclass exists that has a membership degree greater than the membership degree to which it belongs to the superclass, even if this membership degree is very small. To avoid propagating infinitesimal degrees, the given threshold is used as a computational threshold (the same convention is adopted in the rest of the paper).

Formally, let c' and c'' be (fuzzy) classes and β be a given threshold. We say that c'' is a subclass of c' if

$$(\forall o)(\beta \le \mu_{c''}(o) \le \mu_{c'}(o)).$$

The membership degree to which c'' is a subclass of c' should be $\min_{\mu c''(o) \ge \beta}(\mu_{c''}(o))$. Here, o is an object instance of c' and c'' in the universe of discourse, and $\mu_{c'}(o)$ and $\mu_{c''}(o)$ are membership degrees of o to c' and c'', respectively.

However, note that in the above-mentioned fuzzy generalization relationship we assume that classes c' and c'' only have the second level of fuzziness. Classes c' or c'' may be classes with membership degrees, namely, with the first level of fuzziness. Assume that we have two classes c' and c'', as follows:

c' WITH *degree_c'* DEGREE,

c'' WITH *degree_c''* DEGREE.

Then, c'' is a subclass of c' if

$$(\forall o)(\beta \le \mu_{c''}(o) \le \mu_{c'}(o)) \wedge ((\beta \le degree\_c'' \le degree\_c').$$

That means that c'' is only a subclass of c' if the membership degrees of all objects to c' and c'' are greater than or equal to the given threshold and the membership degree of any object to c' is greater than or equal to the membership degree of this object to c'', the membership degrees of c' and c'' are greater than or equal to the given threshold, and the membership degree of c' is greater than or equal to the membership degree of c''.

Consider a fuzzy superclass c' and its fuzzy subclasses $c''_1, c''_2, \ldots, c''_n$ with instance membership degrees $\mu_{c'}$, $\mu_{c''1}$, $\mu_{c''2}, \ldots$, and $\mu_{c''n}$, respectively, which also have membership degrees of *degree_c'*, *degree_c''*$_1$, *degree_c''*$_2$, …, and *degree_c'*$_n$, respectively. Then, the following relationship is true:

$$(\forall o)(\max(\mu_{c''1}(o), \mu_{c''2}(o), \ldots, \mu_{c''n}(o)) \leq \mu_{c'}(o)) \wedge (\max(degree\_c''_1, degree\_c''_2, \ldots, degree\_c''_n) \leq degree\_c').$$

For a crisp superclass/subclass relationship with multiple subclasses, an object must belong to the superclass if it belongs to one subclass, but an object belonging to the superclass may or may not belong to the subclasses. Consider the classes *Patient*, *Outpatient* and *Inpatient*, where *Outpatient* and *Inpatient* are two subclasses of *Patient*. Suppose that the object instance *Michael* is an object instance of the subclass *Inpatient*. Clearly, *Michael* must be an object instance of the superclass *Patient*. Now, suppose that *Michael* is an object instance of the superclass *Patient* but not of subclass *Outpatient*, which implies that in the superclass/subclass relationship, the instance membership degree to which an object belongs to the subclasses is not greater than the instance membership degree to which this object belongs to the superclasses. Consequently, in the fuzzy superclass/subclass relationship with multiple subclasses, the instance membership degree to which an object belongs to any of the subclasses is not greater than the instance membership degree to which this object belongs to the superclasses. Accordingly, the *max* operator is used above.

We assess the above fuzzy subclass/superclass relationships by utilizing the degree of inclusion of objects to the class. Clearly, such an assessment is based on the extensional viewpoint of the class. When classes are defined with the intensional viewpoint, there is no object available. Therefore, the method given above cannot be used. At this point, we have to use the degree of inclusion of a class with respect to another class to determine the relationships between the fuzzy subclass and superclass. The notion of inclusion degree was originally developed in Ma et al. [31] to assess data redundancy in fuzzy-relational databases. Ma et al. [31] extended the inclusion degree to evaluate the membership degree of an object to a class and to further develop the relationship between fuzzy subclass and superclass.

Formally, let c' and c'' be (fuzzy) classes and the degree to which c'' is the subclass of c' be denoted by $\mu(c', c'')$. For a given threshold β, we say that c'' is a subclass of c' if

$$\mu(c', c'') \geq \beta.$$

Here $\mu(c', c'')$ is used to evaluate the inclusion degree of c'' with respect to c' according to the inclusion degree of the attribute domains of c'' with respect to the attribute domains of c' as well as the weights of the attributes. The membership degree to which c'' is a subclass of c' is $\mu\,(c', c'')$.

Now, consider the situation in which classes c' or c'' are classes with membership degrees, namely, with the first level of fuzziness. Assume that we have two classes c' and c'', as follows:

c' WITH *degree_c'* DEGREE,

c'' WITH *degree_c''* DEGREE.

Then, c'' is a subclass of c' if

$$(\mu(c', c'') \geq \beta) \wedge (\beta \leq degree\_c'' \leq degree\_c').$$

This means that c'' is only a subclass of c' if the inclusion degree of c' with respect to c'' is greater than or equal to the given threshold, the membership degrees of c' and c'' are all greater than or equal to the given threshold, and the membership degree of c' is greater than or equal to the membership degree of c''.

The inclusion degree of a (fuzzy) subclass with respect to the (fuzzy) superclass can be calculated according to the inclusion degree of the attribute domains of the subclass with respect to the attribute domains of the superclass as well as the weights of attributes. The methods used to evaluate the inclusion degree of fuzzy attribute domains and to further evaluate the inclusion degree of a subclass with respect to the superclass were developed in Ma et al. [31]. Note that this study did not discuss the relationship between a subclass and a superclass with the first level of fuzziness.

A critical issue in subclass/superclass hierarchies is the multiple inheritances of a class. The fuzzy generalization relationship with multiple inheritances of class can be interpreted according to a conjunctive or disjunctive interpretation. Let c', c_1 and c_2 be fuzzy classes with the second level of fuzziness and β be a given threshold. We say that c' is a

subclass of c_1 and c_2 with a conjunctive interpretation if

$$(\forall o)(\forall c)(c \in \{c_1, c_2\} \wedge \beta \leq \mu_{c'}(o) \leq \mu_c(o)).$$

If instead of using the inclusion degree of objects to the class we use the inclusion degree of a class with respect to another class to determine the relationships between fuzzy subclass and superclass, the above formula is redefined as follows:

$$(\mu(c_1, c') \geq \beta) \wedge (\mu(c_2, c') \geq \beta).$$

Now, suppose that c', c_1 and c_2 above may also have the first level of fuzziness, with membership degrees *degree_c'*, *degree_c*$_1$ and *degree_c*$_2$, respectively. Then, we have

$$(\forall o)(\forall c)(c \in \{c_1, c_2\} \wedge \beta \leq \mu_{c'}(o) \leq \mu_c(o) \wedge \beta \leq degree\_c' \leq \min(degree\_c_1, degree\_c_2)).$$

If we use the inclusion degree of a class with respect to another class to determine the relationships between fuzzy subclasses and the superclass, the above formula is redefined as follows:

$$(\mu(c_1, c') \geq \beta) \wedge (\mu(c_2, c') \geq \beta) \wedge (\beta \leq degree\_c' \leq degree\_c_1) \wedge \beta \leq degree\_c' \leq degree\_c_2).$$

In instances in which a class has multiple inheritances, ambiguity arises when more than one of the superclasses have common attributes and the subclass does not declare explicitly the class from which the attribute is inherited. Assume that the attribute a_i in c_1, denoted by $a_i(c_1)$, is the same as the attribute a_i in c_2, denoted by $a_i(c_2)$. If $a_i(c_1)$ and $a_i(c_2)$ have an identical domain, then there is no conflict in the multiple inheritance hierarchy and c inherits attribute a_i directly. However, a conflict occurs if $a_i(c_1)$ and $a_i(c_2)$ have different domains. At this point, whether c inherits $a_i(c_1)$ or $a_i(c_2)$ depends on which is dominant [31], with class c inheriting a_i from the dominant superclass. Note that in a fuzzy multiple inheritance hierarchy, the subclass has different degrees with respect to different superclasses, which is different from the situation in the classical object-oriented databases.

Now, consider the fuzzy generalization relationship with a disjunctive interpretation. Let c', c_1 and c_2 be fuzzy classes with the second level of fuzziness and β be a given threshold. We say that c' is a subclass of c_1 and c_2 with a disjunctive interpretation if

$$(\forall o)(\exists c)(c \in \{c_1, c_2\} \wedge \beta \leq \mu_{c'}(o) \leq \mu_c(o)).$$

If, instead of using the inclusion degree of objects to the class, we use the inclusion degree of a class with respect to another class to determine the relationships between a fuzzy subclass and its superclasses, the above formula is redefined as follows:

$$(\mu(c_1, c') \geq \beta) \vee (\mu(c_2, c') \geq \beta).$$

Now suppose that c', c_1 and c_2 above may also have the first level of fuzziness with membership degrees *degree_c'*, *degree_c*$_1$ and *degree_c*$_2$, respectively. Then, we have

$$(\forall o)(\exists c)(c \in \{c_1, c_2\} \wedge \beta \leq \mu_{c'}(o) \leq \mu_c(o) \wedge \beta \leq degree\_c' \leq degree\_c).$$

If we use the inclusion degree of a class with respect to another class to determine the relationships between a fuzzy subclass and superclass, the above formula is redefined as follows:

$$(\mu(c_1, c') \geq \beta \wedge \beta \leq degree\_c' \leq degree\_c_1) \vee (\mu(c_2, c') \geq \beta \wedge \beta \leq degree\_c' \leq degree\_c_2).$$

A dashed triangular arrowhead is used to represent a fuzzy generalization as shown in Fig. 2. Here, the classes *Young Employee*, *Middle-Aged Employee* and *Old Employee* all exhibit the second level of fuzziness, meaning that the classes may have some instances (objects) that belong to the classes with membership degrees. These three classes can be generalized into the class *Employee*.

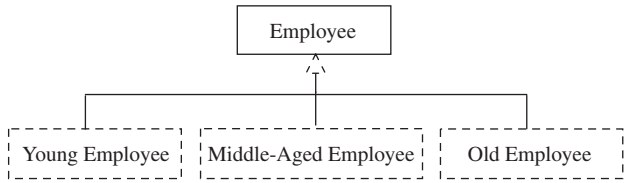

Fig. 2. A fuzzy generalization relationship.

3.3. Fuzzy aggregation

An aggregation captures a whole-part relationship between an aggregate (the whole) and several constituent parts (the part), where the constituent parts can exist independently. Each instance of an aggregate can be projected into a set of instances of constituent parts. Formally, let c' be an aggregation of constituent parts $c''_1, c''_2, \ldots,$ and c''_n. For $o \in c'$, the projection of o to c''_i is denoted by $o \downarrow_{c''i}$, which represents an instance of c''_i. Then, we have $(o \downarrow_{c''1}) \in c''_1, (o \downarrow_{c''2}) \in c''_2, \ldots, (o \downarrow_{c''n}) \in c''_n$. For example, aggregate class *Car* is aggregated by the constituent part classes *Engine*, *Interior* and *Chassis*. For a car instance (such as "*Honda CR-V EX*"), its projection on *Engine* is an engine instance (say "*In-Line 4-Cylinder*"), and we have Honda *CR-V EX* $\downarrow_{Engine}$= *In-Line 4-Cylinder*.

A class aggregated from fuzzy constituent parts may be fuzzy. If the former is still called an aggregate, then the aggregation is a fuzzy aggregation. At this point, a class is an aggregation of constituent parts with membership degrees of [0, 1]. Correspondingly, the following method can be used to determine the fuzzy aggregation relationship.

(a) For any (fuzzy) object, the membership degree to which it belongs to the aggregate is greater than or equal to the given threshold, and
(b) The membership degree to which it belongs to the aggregate is less than or equal to the membership degree to which its projection to each constituent part belongs to the corresponding constituent part.

The aggregate is then an aggregation of the constituent parts with membership degrees, which is the minimum of the membership degrees to which the projections of these objects to the constituent parts belong to the corresponding constituent parts. As for the fuzzy generalization, the given threshold is a computational threshold that is used to set the confidence to which the aggregate and constituent parts have an aggregation relationship with a membership degree and to avoid propagating infinitesimal degrees. The objects of the aggregate that have membership degrees greater than or equal to the threshold are tested for whether their membership degrees are less than or equal to the membership degrees to which their projections to each constituent part belong to the corresponding constituent part. The objects of the aggregate that have membership degrees less than the threshold are not considered.

Formally, let c' be a fuzzy aggregation of fuzzy class sets $c''_1, c''_2, \ldots,$ and c''_n, with corresponding instance membership degrees of $\mu_{c'}, \mu_{c''1,}\mu_{c''2,} \ldots,$ and $\mu_{c''n}$, respectively. Let β be a given threshold. Then,

$$(\forall o)(o \in c' \wedge \beta \leq \mu_{c'}(o) \leq \min(\mu_{c''1}(o \downarrow_{c''1}), \mu_{c''2}(o \downarrow_{c''2}), \ldots, \mu_{c''n}(o \downarrow_{c''n}))).$$

Therefore, a fuzzy class c' is the aggregate of any group of fuzzy classes $c''_1, c''_2, \ldots,$ and c''_n if, for any (fuzzy) object instance, the membership degree to which it belongs to class c' is less than or equal to the member degree to which its projection to any one of $c''_1, c''_2, \ldots,$ and c''_n, e.g., $c''_i (1 \leq i \leq n)$, belongs to class c''_i. For any (fuzzy) instance object, the membership degree to which it belongs to class c' must be greater than or equal to the given threshold. The membership degree to which c' is an aggregation of class sets $c''_1, c''_2, \ldots,$ and c''_n should be $\min_{\mu c''i(e \downarrow c''i) \geq \beta}(\mu_{c''i}(o \downarrow_{c''i}))(1 \leq i \leq n)$. Here, o is an object instance of c'.

For a classical aggregation, for any instance of aggregate c', its projection to any one of $c''_1, c''_2, \ldots,$ and c''_n, e.g., $c''_i (1 \leq i \leq n)$, must be the object instance of the constituent part c''_i. However, an instance of constituent part c''_i may or may not be used to constitute the object instance of aggregate c' as one part because the constituent part exists independently. This implies that in the fuzzy aggregation, the instance membership degree to which an object belongs to the aggregate is not greater than the instance membership degree to which the projection of this object on any one of the constituent parts belongs to the corresponding constituent part. Consequently, in the fuzzy aggregation, the instance membership degree to which an object belongs to the aggregate is not greater than the instance membership degree

projected by this object on any one of the constituent parts belonging to the corresponding constituent part. Therefore, a min operator is adopted in the fuzzy aggregation above.

Now consider the first level of fuzziness in the above-mentioned classes c', c''_1, c''_2, …, and c''_n, namely, that they are fuzzy classes with membership degrees. Let

c' WITH *degree_c'* DEGREE,

c''_1 WITH *degree_c''*$_1$ DEGREE,

c''_2 WITH *degree_c''*$_2$ DEGREE,

……

c''_n WITH *degree_c''*$_n$ DEGREE.

Then, c' is an aggregate of c''_1, c''_2, …, and c''_n if

$$(\forall o)(o \in c' \wedge \beta \le \mu_{c'}(o) \le \min(\mu_{c''1}(o \downarrow_{c''1}), \mu_{c''2}(o \downarrow_{c''2}), \ldots, \mu_{c''n}(o \downarrow_{c''n})) \wedge degree\_c' \le \min(degree\_c''_1, degree\_c''_2, \ldots, degree\_c''_n)).$$

Here, β is a given threshold.

Note that the assessment of the fuzzy aggregation relationships given above is based on the extensional viewpoint of class. Clearly, these methods cannot be used if the classes are defined with the intensional viewpoint, because in that case no object would be available. Below we discuss how to determine the fuzzy aggregation relationship using the inclusion degree.

Formally, let c' be a fuzzy aggregation of fuzzy classes c''_1, c''_2, …, and $c''_{n,}$ and β be a given threshold. Also, let the projection of c' to c''_i be denoted by $c' \downarrow_{c''i}$. Then,

$$\min(\mu(c''_1, c' \downarrow_{c''1}), \mu(c''_2, c' \downarrow_{c''2}), \ldots, \mu(c''_n, c' \downarrow_{c''n})) \ge \beta.$$

Here, $\mu(c''_i, c' \downarrow_{c''i})(1 \le i \le n)$ indicates the degree to which c''_i semantically includes $c' \downarrow_{c''i}$. The membership degree to which c' is an aggregation of c''_1, c''_2, …, and c''_n is $\min(\mu(c''_1, c' \downarrow_{c''1}),\ \mu(c''_2, c' \downarrow_{c''2}), \ldots, \mu(c''_n, c' \downarrow_{c''n}))$.

Furthermore, the expression above can be extended for the situation that c', c''_1, c''_2, …, and c''_n may exhibit the first level of fuzziness, namely, that they may be fuzzy classes with membership degrees. Let β be a given threshold and

c' WITH *degree_c'* DEGREE,

c''_1 WITH *degree_c''*$_1$ DEGREE,

c''_2 WITH *degree_c''*$_2$ DEGREE,

……

c''_n WITH *degree_c''*$_n$ DEGREE.

Then, c' is an aggregate of c''_1, c''_2, …, and c''_n if

$$\min(\mu(c''_1, c' \downarrow_{c''1}), \mu(c''_2, c' \downarrow_{c''2}), \ldots, \mu(c''_n, c' \downarrow_{c''n})) \ge \beta \wedge degree\_c' \le \min(degree\_c''_1, degree\_c''_2, \ldots, degree\_c''_n)).$$

A dashed open diamond is used to denote a fuzzy aggregate relationship as shown in Fig. 3 in which a car is aggregated from the engine, interior, and chassis. In Fig. 3, the engine is old, so the fuzzy class *Old Engine* exhibits the second level of fuzziness. The class *Old Car* is aggregated from the classes *interior* and *chassis* and the fuzzy class *old engine*, making *Car* a fuzzy class with the second level of fuzziness.

3.4. Fuzzy association

An association relationship with an association name is defined as a binary structural relationship between two classes that associates them, which specifies that the objects of one class are connected to the objects of another class.

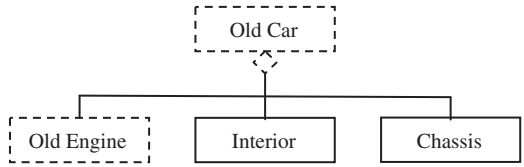

Fig. 3. A fuzzy aggregation relationship.

An association relationship is bidirectional or unidirectional. The association relationship is defined on the basis of classes, not on the basis of the objects of classes. Of course, for a given scope, each pair of objects in the corresponding classes has the same association relationship. In an airport, for example, the classes *Flight* and *Plane* have an association relationship with the association name *assignedPlane*.

Fuzzy classes are associated and then constitute the fuzzy association relationship. For fuzzy classes with the second level of fuzziness, the class instances belong to the given classes with membership degrees. As a result, it is possible for it to be unknown for certain whether two class instances belonging to the associated classes exhibit the given association relationship, although the association relationship definitely exists between these two classes. For example, the classes *Young Driver* and *New Car* have such a fuzzy association relationship with the association name *driving*. Here, an instance of *Young Driver* and an instance of *New Car* belong to the corresponding classes with membership degrees, and thus these two instances have an association relationship with a membership degree.

Formally, let c' and c'' be two classes with the second level of fuzziness. The instance $o\prime$ of c' is an object with the membership degree $\mu_{c'}(o')$, and the instance o'' of c'' is an object with the membership degree $\mu_{c''}(o'')$. Assume that the association relationship between c' and c'' is denoted by *assc* (c', c''). It is clear that the association relationship between o' and o'', denoted *assc* (o', o''), has a membership degree, which can be calculated by

$$\mu_{(assc(o',o''))} = \min(\mu_{c'}(o'), \mu_{c''}(o'')).$$

Note that the membership degree $\mu_{(assc(o',o''))}$ is defined only for o' and o'', not for c' and c''. In other words, the membership degree to which o' and o'' have the association relationship is $\mu_{(assc(o',o''))}$. For a pair of instances that belong to the two classes and are not o' and o'', the membership degree to which the pair of instances has the association relationship may be different from $\mu_{(assc(o',o''))}$. This pair of instances is not regarded as having the association relationship if the membership degree is small enough.

Note also that the association relationship may be fuzzily defined (e.g., People *like* Sports) and that the association relationship exists in two associated classes with a membership degree. Different from the fuzzy association relationship at the class instance level above, the fuzzy association relationship here is at the class level. This level of fuzziness in the association relationship can be indicated explicitly by the designers even if the corresponding classes are crisp. In the preliminary design of a product car, for example, suppose that a *DVD player* may or may not be installed in the car at this stage, and the possibility that the DVD player will be installed in the car is 0.6. Let c' and c'' be two crisp classes and *assc* (c', c'') is the association relationship with the membership degree *degree_assc*, denoted *assc* (c', c'') WITH *degree_assc* DEGREE. At this moment, for any instance o' of c' and any instance o'' of c'', $\mu_{c'}(o') = 1.0$ and $\mu_{c''}(o'') = 1.0$. Then, we have

$$\mu_{(assc(c',c''))} = degree\_assc \text{ and } \mu_{(assc(o',o''))} = degree\_assc.$$

Here, $\mu_{(assc(c',c''))}$ is used to represent the membership degree to which c' and c'' have the association relationship, and $\mu_{(assc(o',o''))}$ is used to represent the membership degree to which o' and o'' have the association relationship.

The classes with the first level of fuzziness generally result in a fuzzy association relationship with a membership degree if this association is not indicated explicitly. Formally, let c' and c'' be two classes with only the first level of fuzziness, denoted c' WITH *degree_c'* DEGREE and c'' WITH *degree_c''* DEGREE, respectively. Then, the association relationship between c' and c'', denoted *assc* (c', c''), has the membership degree *degree_assc*, denoted *assc* (c', c'') WITH *degree_assc* DEGREE. Here, *degree_assc* is calculated by

$$degree\_assc = \min(degree\_c', degree\_c'').$$

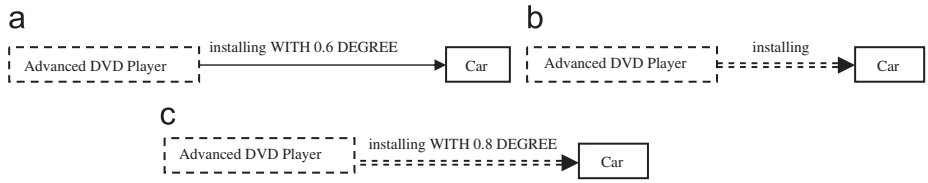

Fig. 4. Fuzzy association relationships.

For any instance o' of c' and any instance o'' of c'', ($\mu_{c'}(o') = 1.0$ and $\mu_{c''}(o'') = 1.0$. We have

$$\mu_{(assc(c',c''))} = degree\_assc = \min(degree\_c', degree\_c'') \quad \text{and}$$

$$\mu_{(assc(o',o''))} = degree\_assc = \min(degree\_c', degree\_c'').$$

Finally, consider the situation in which the classes exhibit the first and second levels of fuzziness. On one hand, the two classes have a fuzzy association relationship at the class level. On the other hand, the class instances of these two classes may have fuzzy association relationships at the class instance level. Let c' and c'' be two classes with the first level of fuzziness, denoted c' WITH $degree\_c'$ DEGREE and c'' WITH $degree\_c''$ DEGREE, respectively. Also, let the instance o' of c' have the membership degree $\mu_{c'}(o')$, and the instance o'' of c'' have the membership degree $\mu_{c''}(o'')$.

Let $assc\ (c', c'')$ be the association relationship with membership degree between c' and c'', and let $assc\ (o', o'')$ be the association relationship with membership degree between o' and o''. Then, we have

$$\mu_{(assc(c',c''))} = \min(degree\_c', degree\_c'') \quad \text{and}$$

$$\mu_{(assc(o',o''))} = \min(\mu_{c'}(o'), \mu_{c''}(o''), degree\_c', degree\_c'').$$

The pair of words WITH *mem* DEGREE ($0 \leq mem \leq 1$) after the association name of an association relationship represents the association relationship with membership degree. We use a double line with an arrowhead to denote the association relationship by which the class instances are associated.

Fig. 4 shows two kinds of fuzziness in the fuzzy association relationships. In (a), classes *Advanced DVD Player* and *Car* have an association relationship *installing* with a 0.6 membership degree. Also, it is possible that the DVD player will certainly be installed in the car. In this case, the possibility that the DVD player will be installed in the car is 1.0, and the classes *Advanced DVD Player* and *Car* have an association relationship *installing* with a 1.0 membership degree. As shown in (b), at the level of instances, the instances of classes *Advanced DVD Player* and *Car* may or may not have the association relationship *installing.* In (c), two kinds of fuzzy association relationship in (a) and (b) arise simultaneously.

3.5. Fuzzy dependency

A dependency signifies a supplier/client relationship between model elements, where the modification of the supplier may impact the client model elements. That means the client is not complete without the supplier. A dependency relationship is different from an association relationship in that it is only unidirectional. A dependency relationship between the supplier class and the client class is established on the basis of classes, not on the basis of the objects of classes. The dependency relationship is only related to the classes themselves and does not require a set of instances to give it meaning. Each pair of objects where each one belongs to the corresponding class exhibits the dependency relationship. For example, the supplier class *Employee* and the client class *Employee Dependents* have a dependency relationship.

For fuzzy classes with the second level of fuzziness, the dependency relationship is not affected by such fuzzy classes because the dependency relationship is defined at the level of classes over classes. However, the class instances belong to the given classes with membership degrees. Therefore, two class instances that belong to the related classes may

Fig. 5. Fuzzy dependency relationship.

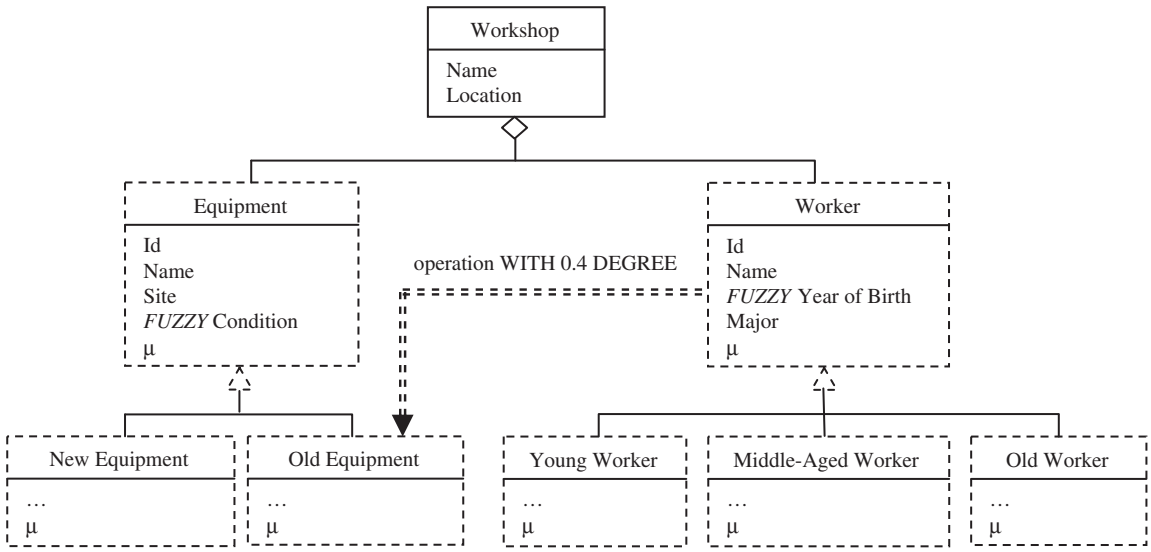

Fig. 6. A fuzzy UML data model.

have a dependency relationship with a membership degree. Note that the membership degree here is only for the given class instances, not for the corresponding classes.

Like the fuzzy association relationship above, the dependency relationship between classes may be a fuzzy one with a membership degree at the level of classes, which can be indicated explicitly by the designers or implied implicitly by the source class based on the fact that the client class is decided by the supplier class. Assume that the supplier class is fuzzy with the first level of fuzziness. Then, the client class must be fuzzy with the first level of fuzziness. The membership degree of the client class is determined by the supplier class as the possibility degree of the supplier class.

For the supplier class *Employee WITH 0.85 DEGREE*, for example, the client class *Employee Dependent* should be *Employee Dependent WITH 0.85 DEGREE*. The dependency relationship between *Employee* and *Employee Dependent* should be fuzzy with a 0.85 membership degree. Note that the fuzziness of the dependency relationship among the classes, if not given explicitly, is implied by first level of fuzziness of the supplier class. Because the fuzziness of the dependency relationship is denoted implicitly by the first level of fuzziness of the supplier class, a dashed line with an arrowhead can still be used to denote the fuzziness in the dependency relationship. Fig. 5 shows a fuzzy association relationship at the level of classes.

3.6. An example

Employing some notations introduced above, we now present a fuzzy UML data model of a workshop in Fig. 6. In this example, a *Workshop* is simply aggregated by two classes, *Equipment* and *Worker*. The class *Equipment* is a superclass with *New Equipment* and *Old Equipment* as its two fuzzy subclasses in that they may have fuzzy instances. Similarly, the class *Worker* has three fuzzy subclasses: *Young Worker*, *Middle-Aged Worker* and *Old Worker*. The classes *Worker* and *Old Equipment* have the fuzzy association relationship *operation*, which exhibits fuzziness at both the first and second levels. The class *Worker* has four attributes, with *Id*, *Name* and *Major* taking on crisp values, and the attribute *Year of Birth* taking on a fuzzy value. Also, the class *Equipment* has four attributes, with *Id*, *Name* and *Site* taking on crisp values and the attribute *Condition* taking on a fuzzy value.

4. Mapping of the fuzzy UML data model into the fuzzy object-oriented database model

4.1. Fuzzy object-oriented database (FOODB) model

Ma et al. [31] developed a fuzzy object-oriented database (FOODB) model in which the classes may be fuzzy. Accordingly, in the FOODB, an object belongs to a class with a membership degree of [0, 1] and a class is the subtype of another class with a degree of [0, 1]. In the FOODB, the specification of a class includes the definition of ISA relationships, attributes and method implementations. Some additional definitions are needed to specify a fuzzy class. First, weights must be assigned to the attributes of the class. In addition to these common attributes, a new attribute should be added to the class to indicate the membership degree to which an object belongs to the class. If a class is a fuzzy subclass, its superclasses and the degree to which the class is the subclass of the superclasses should be illustrated in the specification of the class. Finally, in the definition of a fuzzy class, fuzzy attributes are explicitly indicated.

Formally, a fuzzy class is defined as follows:

CLASS *class name* WITH DEGREE OF *degree*
  INHERITS *supertype_1 name* WITH DEGREE OF *degree_1*
  …
  INHERITS *supertype_k name* WITH DEGREE OF *degree_k*
  ATTRIBUTES
    Attribute_1 name: [FUZZY] DOMAIN *dom_1*: TYPE OF *type_1* WITH DEGREE OF *degree_1*
    …
    Attribute_m name: [FUZZY] DOMAIN *dom_m*: TYPE OF *type_m* WITH DEGREE OF *degree_m*
    Membership_Attribute name: *membership_degree*
  WEIGHT
    w (*Attribute*_1 *name*) = *w*_1
    …
    w (*Attribute_m name*) = *w_m*
  METHODS
    …
END.

4.2. Transformation of fuzzy UML class diagrams

Fuzzy information can be modeled in the UML data model and the object-oriented data model using the fuzzy UML data model and the FOODB, respectively. This section investigates the mapping of the fuzzy UML data model into the FOODB schema.

Generally speaking, the classes in the UML data model correspond to classes in the object-oriented database schema. The attributes in a class of the UML data model correspond to the attributes of the class of the OODB schema. The generalization in UML closely conforms to subclass/superclass structures, and the transformations are easily conducted. In the fuzzy UML data model, the class and generalization may be fuzzy. Following similar transformation rules as above, we develop a formal approach to transform a fuzzy UML data model into a FOODB model.

4.2.1. Transformation of classes

The classes in the UML data model generally correspond to the classes in the object-oriented database schema, and the attributes of the classes in the UML data model correspond to the attributes of the classes in the object-oriented database schema. If the classes in the UML data model are subclasses or superclasses in the object-oriented database schema, the inheritance hierarchies of the classes produced by these UML classes must be explicitly indicated.

To transform from the fuzzy UML data model into the fuzzy object-oriented database schema, we first suppose that the classes in the UML data model are neither subclasses nor superclasses. Then, we can distinguish four basic kinds of classes in the fuzzy UML model as follows.

(a) classes without any fuzziness;
(b) classes with fuzziness only at the third level;
(c) classes with fuzziness only at the second level;
(d) classes with fuzziness only at the first level.

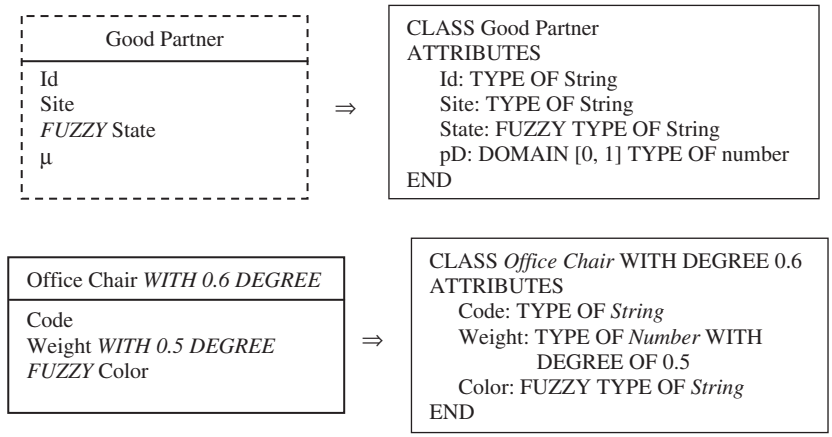

Fig. 7. Transformation of the classes in the fuzzy UML to the fuzzy object-oriented database scheme.

When transforming the classes of the fuzzy UML data model into the classes of the fuzzy object-oriented database schema and transforming the attributes of the former classes into the attributes of the later classes, the transformation of the classes with three levels of fuzziness is of particular concern. For the classes in case (b), the attributes taking fuzzy values have fuzzy value types denoted by *FUZZY TYPE OF* in the transformed classes in the fuzzy object-oriented database schema. For the classes in case (c), an additional attribute denoted by *pD* should be added into each class transformed from the corresponding class in the fuzzy UML data model, which is used to denote the possibility that the class instances belong to the class. For the classes in case (d), the classes and/or their attributes may be associated with membership degrees. Correspondingly, the transformed classes and attributes in the fuzzy object-oriented database schema are associated with membership degrees, if any. The membership degree is used to indicate the possibility that the created class belongs to the corresponding database schema or that the attributes of the created class belong to the class.

Based on the three basic types of fuzzy classes above (i.e., cases (b)–(d)), several combined types of fuzzy classes can be constructed. For classes with fuzziness both at the third and second levels (denoted (e)), their transformation is a simple combination of the above transformations in cases (b) and (c). For the classes with fuzziness both at the third and first levels (denoted (f)), their transformation is a simple combination of the above transformations in cases (b) and (d). For the classes with fuzziness both at the second and first levels (denoted (g)), their transformation is a simple combination of the above transformations in cases (c) and (d). For the classes with fuzziness at all three levels (denoted (h)), their transformation is a simple combination of the above transformations in cases (b)–(d).

Fig. 7 shows the transformation of the classes in the fuzzy UML data models to the fuzzy object-oriented database schema. For the sake of simplification, some components in the class definition of the fuzzy object-oriented schema such as *DOMAIN*, *WEIGHT*, *METHODS*, and *CONSTRAINTS* are not listed.

Assume that the classes of the fuzzy UML data model are superclasses that may belong to any of the class types listed in cases (a)–(h) above. The transformation of such classes into the classes of the fuzzy object-oriented database schema is the same as the transformation of the classes of the fuzzy UML data model given above. The classes of the fuzzy UML data model that are subclasses and may be of any type (a)–(h) can be transformed into the classes of the fuzzy object-oriented database schema following the same principles of the class transformation given above. However, in the fuzzy object-oriented database schema, the inheritance hierarchies of the produced classes (subclasses) must be explicitly indicated.

Fig. 8 shows the transformation of the subclasses in the fuzzy UML data models to the fuzzy object-oriented database schema.

4.2.2. Transformation of aggregations

An aggregation specifies a whole-part relationship between constituent parts and an aggregate that is a class representing the whole. In the fuzzy UML data model, the (fuzzy) aggregate can be transformed into a class in the fuzzy

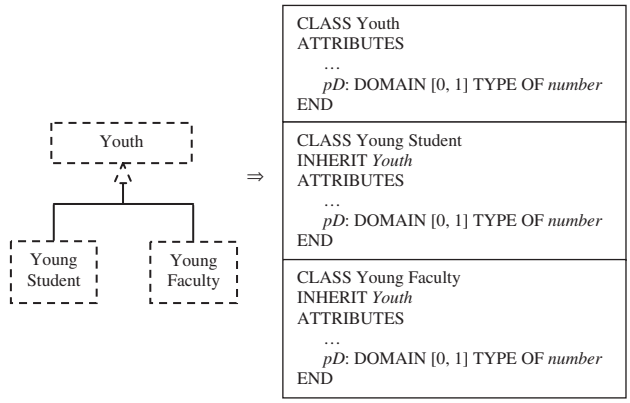

Fig. 8. Transformation of the subclasses in the fuzzy UML to the fuzzy object-oriented database scheme.

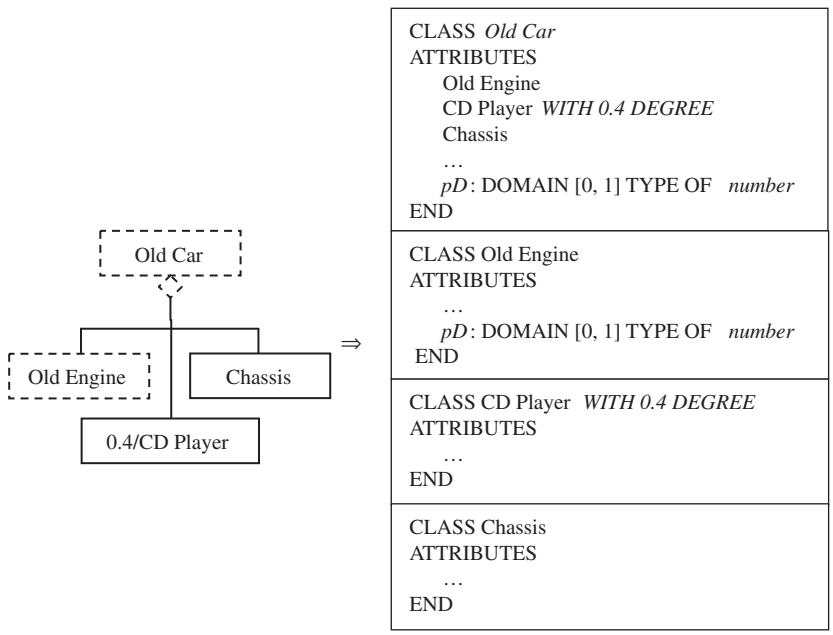

Fig. 9. Transformation of the aggregations in the fuzzy UML to the fuzzy object-oriented database scheme.

object-oriented database schema, called an *aggregation class*, according to the transformation of the classes given above. Also, each constituent part, being a (fuzzy) class, can be transformed into a class in the fuzzy object-oriented database schema, called a *component class*. Note that the attributes of the aggregation class consist of all attributes from the aggregate as well as all component classes as complex class attributes.

Fig. 9 shows the transformation of the aggregations in the fuzzy UML data models to the fuzzy object-oriented database schema.

4.2.3. Transformation of associations

An association in the UML data model should be transformed into an association in the OO schema, which describes a pointer as the attribute(s) in a class that combine(s) an explicit reference to another class. Considering the constraint of cardinality, such attribute(s) in two associated classes can be single-valued or multi-valued. In the fuzzy UML data

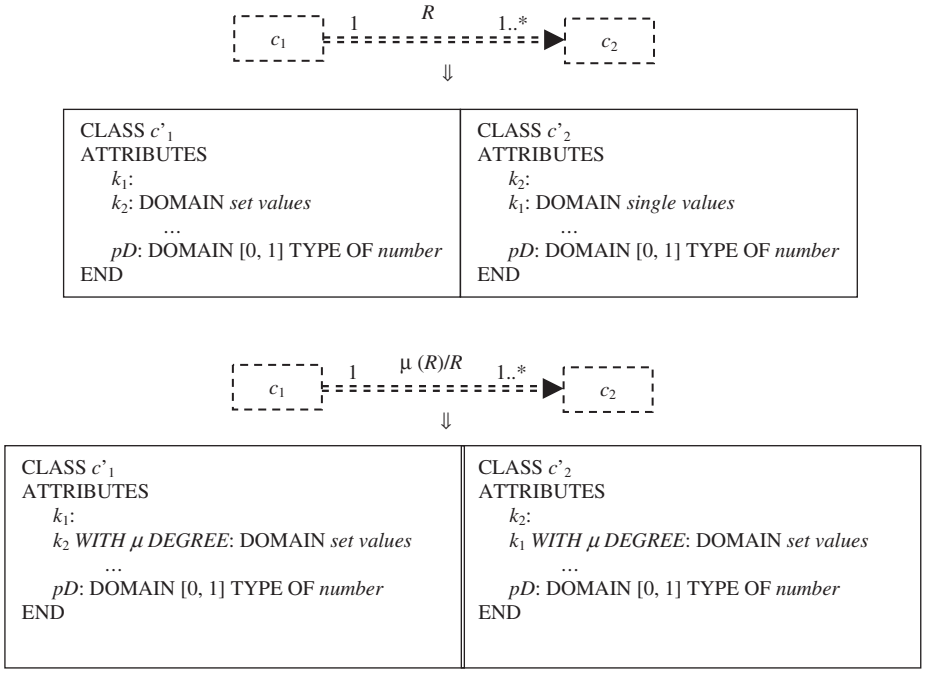

Fig. 10. Transformation of the associations in the fuzzy UML to the fuzzy object-oriented database schema.

model, three basic kinds of associations can be distinguished, as follows:

(a) associations without any fuzziness;
(b) associations with fuzziness only at the second level;
(c) associations with fuzziness only at the first level.

Let class c_1 and class c_2 be connected with the association relationship R, where R may be a one-to-one, one-to-many or many-to-many relationship. Also, assume that c_1 and c_2 have attributes k_1 and k_2, respectively, with values serving as the object identification. Then, following the transformation process of the classes given above, c_1 and c_2 can be transformed into the classes in the fuzzy object-oriented database schema. However, the influence of R on the association of c_2 (or c_1) to c_1 (or c_2) must be considered when c_1 (or c_2) is transformed to a class of the fuzzy object-oriented database schema. Here, k_2 (or k_1) should be added into the class c'_1 (or c'_2) created by c_1 (or c_2) as a foreign key, just like the relational databases. If the constraint of cardinality is a one-to-one relationship, then R is a one-to-one relationship from c_1 to c_2, k_2 is a single-valued attribute in c'_1 and k_1 is a single-valued attribute in c'_2. However, if R is a one-to-many relationship from c_1 to c_2, then k_2 is a multi-valued attribute in c'_1 and k_1 is a single-valued attribute in c'_2. Clearly, k_2 is a multi-valued attribute in c'_1 and k_1 is a multi-valued attribute in c'_2 if R is a many-to-many relationship from c_1 to c_2.

For relationship R in case (b), the possibility that a relationship instance belongs to R should be mapped into the possibility that an object instance belongs to c'_1 or c'_2. Therefore, additional attributes denoting the possibility that the class instances belong to the corresponding class should be added to c'_1 and c'_2, respectively. For relationship R in case (c), i.e., the relationships with the membership degree μ, k_2 in c'_1 and k_1 in c'_2 should be the attributes with membership degrees, indicating the possibility that the foreign key is included in the created class.

Following the above transformations in cases (b) and (c), it is not difficult to deal with relationship R in the case of associations with fuzziness both at the first and second levels, with the transformation carried out as a simple combination of the transformations in cases (b) and (c).

Fig. 10 shows the transformation of the associations.

Note that we only consider an isolated (fuzzy) association in the transformation of the association. That is, we assume that neither of the two associated classes is associated with other classes in addition to the given association relationship. In real applications, however, it is possible that one class may be associated with more than one class.

For example, the classes *Young Person* and *New Car* are connected by the association relationship like, while *New Car* and *Manufacturer* are connected with the association relationship made. The class *New Car* is an association class that is associated with the class *Young Person* and also has another association. Formally, let classes c_1 and c_2 be connected with the association relationship R_{12} and classes c_2 and c_3 be connected with the association relationship R_{23}. Assume that in addition to class c_2, classes c_1 and c_3 are not connected with any other classes. Also assume that c_1, c_2 and c_3 have attributes k_1, k_2 and k_3, respectively, the values of which serve as the object identification. Then, c_1 and c_3 are transformed into class c_1' with an additional attribute k_2 and class c_3' with an additional attribute k_2, respectively, in the fuzzy object-oriented database schema. However, c_2 is transformed into class c_2' with two additional attributes k_1 and k_3 in the fuzzy object-oriented database schema.

4.2.4. Transformation of dependencies

In a fuzzy dependency relationship, the client class fuzzily depends on the supplier class. There are several general strategies to transform the (fuzzy) dependency relationship. The first strategy is a view of the independent class in which the client and supplier classes are transformed using the transformation approaches of the classes given above, and the dependency relationship is stated in the transformed classes as a kind of constraint. The second strategy is an aggregation view in which the client class is regarded as a constituent part of the supplier class and is transformed into the complex class attribute of the aggregation class. Here, the aggregation class is transformed from the supplier class according to the transformation approaches of the aggregations given above. The final strategy is a view of association in which the client and supplier classes are similarly transformed using the transformation approaches of the associations given above with a small difference noted below.

Let classes c_1 and c_2 be the supplier and client classes, respectively. Assume that the values of attribute k_1 in c_1 and attribute k_2 in c_2 serve as the object identifiers. Then, c_1 and c_2 can be transformed into classes in the fuzzy object-oriented database schema. However, the dependency of c_2 on c_1 must be considered when c_2 is transformed to a class of the fuzzy object-oriented database schema. Here, k_1 should be added into the class c_2' created by c_2. Note that differently from the transformation of association, k_2 is not added into the class c_1' created by c_1 in this case.

5. Conclusions

This paper presents a fuzzy extended UML data model to cope with fuzzy as well as complex objects in the real world at a conceptual level. Different levels of fuzziness are introduced into the UML class diagram, and the corresponding graphical representations are developed. The classical UML data model is shown to be essentially a subset of the fuzzy UML data model. When no fuzziness exists in the universe of discourse, the fuzzy UML data model can be reduced to the classical UML data model. Also, we formally mapped the fuzzy UML data model into the fuzzy object-oriented database model, including the transformations of the classes, generalizations, aggregations, associations and dependencies in the fuzzy UML data model.

Note that the focus of this paper is on fuzzy data modeling in UML. As we know, UML can be used for knowledge modeling, and knowledge may generally be imprecise and uncertain. In the future work, we will concentrate on the studies of class operations, constraints and rules in fuzzy UML modeling. In addition, we will apply the fuzzy UML data model to Semantic Web ontology.

Acknowledgments

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. This work was supported by the *National Natural Science Foundation of China* (60873010 and 61073139) and *the Fundamental Research Funds for the Central Universities* (N090504005 and N090604012) and in part by the *Program for New Century Excellent Talents in University* (NCET-05-0288).

References

[1] P.P. Chen, The entity-relationship model: toward a unified view of data, ACM Trans. Database Syst. 1 (1) (1976) 9–36.

[2] D.W. Embley, T.W. Ling, Synergistic database design with an extended entity-relationship model, in: Proceedings of the Eight International Conference on Entity-Relationship Approach, 1989, pp. 111–128.

[3] M. Stonebraker, D. Moore, Object-relational DBMSs: the next great wave, Morgan Kaufmann, 1996.
[4] G. Booch, J. Rumbaugh, I. Jacobson, The Unified Modeling Language User Guide, Addison-Welsley Longman, Inc., 1998.
[5] Object Management Group (OMG), Unified Modeling Language (UML), version 1.5, Technical Report, OMG, ⟨www.omg.org⟩, 2003.
[6] D. Berardi, D. Calvanese, G. De Giacomo, Reasoning on UML class diagrams, Artif. Intell. 168 (1–2) (2005) 70–118.
[7] E. Marcos, B. Vela, J.M. Cavero, Extending UML for object-relational database design, Lecture Notes in Computer Science, vol. 2185, 2001, pp. 225–239.
[8] S.W. Ambler, The design of a robust persistence layer for relational databases ⟨http://www.ambysoft.com/persistenceLayer.pdf⟩, 2000.
[9] S.W. Ambler, Mapping objects to relational databases ⟨http://www.AmbySoft.com/mappingObjects.pdf⟩, 2000.
[10] M. Blaha, W. Premerlani, Using UML to design database applications ⟨http://www.therationaledge.com/rosearchitect/mag/archives/9904/f8.html⟩, 1999.
[11] R. Conrad, D. Scheffiner, J.C. Freytag, XML conceptual modeling using UML, in: Proceeding of 19th International Conference on Conceptual Modeling, 2000, pp. 558–571.
[12] K. Falkovych, M. Sabou, H. Stuckenschmidt, UML for the semantic web: transformation-based approaches, in: B. Omelayenko, M. Klein (Eds.), Knowledge Transformation for the Semantic Web, IOS Press, 2003.
[13] M.A. Sicilia, N. Mastorakis, Extending UML 1.5 for fuzzy conceptual modeling: a strictly additive approach, WSEAS Trans. Syst. 5 (3) (2004) 2234–2240.
[14] L.A. Zadeh, Fuzzy sets, Inf. Control 8 (3) (1965) 338–353.
[15] L.A. Zadeh, Fuzzy sets as a basis for a theory of possibility, Fuzzy Sets Syst. 1 (1) (1978) 3–28.
[16] B.P. Buckles, F.E. Petry, A fuzzy representation of data for relational database, Fuzzy Sets Syst. 7 (3) (1982) 213–226.
[17] H. Prade, C. Testemale, Generalizing database relational algebra for the treatment of incomplete or uncertain information and vague queries, Inf. Sci. 34 (1984) 115–143.
[18] K.V.S.V.N. Raju, K. Majumdar, Fuzzy functional dependencies and lossless join decomposition of fuzzy relational database systems, ACM Trans. Database Syst. 13 (2) (1988) 129–166.
[19] M. Umano, S. Fukami, Fuzzy relational algebra for possibility-distribution-fuzzy-relational model of fuzzy data, J. Intell. Inf. Syst. 3 (1994) 7–27.
[20] Z.M. Ma, L. Yan, Updating extended possibility-based fuzzy relational databases, Int. J. Intell. Syst. 22 (3) (2007) 237–258.
[21] G. de Tré, R. de Caluwe, H. Prade, Null values in fuzzy databases, J. Intell. Inf. Syst. 30 (2) (2008) 93–114.
[22] I.J. Blanco, M.A. Vila, C. Martinez-Cruz, The use of ontologies for representing database schemas of fuzzy information, Int. J. Intell. Syst. 23 (4) (2008) 419–445.
[23] R.S. Aygun, A. Yazici, Modeling and management of fuzzy information in multimedia database applications, Multimedia Tools Appl. 24 (1) (2004) 29–56.
[24] J. Chamorro-Martínez, J.M. Medina, C.D. Barranco, E. Galán-Perales, J.M. Soto-Hidalgo, Retrieving images in fuzzy object-relational databases using dominant color descriptors, Fuzzy Sets Syst. 158 (3) (2007) 312–324.
[25] G. Bordogna, G. Pasi, D. Lucarella, A fuzzy object-oriented data model for managing vague and uncertain information, Int. J. Intell. Syst. 14 (1999) 623–651.
[26] V. Cross, R. De Caluwe, N. Van Gyseghem, A perspective from the fuzzy object data management group (FODMG), in: Proceedings of the 1997 IEEE International Conference on Fuzzy Systems, vol. 2, 1997, pp. 721–728.
[27] D. Dubois, H. Prade, J.P. Rossazza, Vagueness, typicality, and uncertainty in class hierarchies, Int. J. Intell. Syst. 6 (1991) 167–183.
[28] R. George, R. Srikanth, F.E. Petry, B.P. Buckles, Uncertainty management issues in the object-oriented data model, IEEE Trans. Fuzzy Syst. 4 (2) (1996) 179–192.
[29] N. Van Gyseghem, R. De Caluwe, Imprecision and uncertainty in UFO database model, J. Am. Soc. Inf. Sci. 49 (3) (1998) 236–252.
[30] J. Lee, N.L. Xue, K.H. Hsu, S.J. Yang, Modeling imprecise requirements with fuzzy objects, Inf. Sci. 118 (1999) 101–119.
[31] Z.M. Ma, W.J. Zhang, W.Y. Ma, Assessment of data redundancy in fuzzy relational database based on semantic inclusion degree, Inform. Process. Lett. 72 (1–2) (1999) 25–29.
[32] N. Marín, O. Pons, M.A. Vila, Fuzzy types: a new concept of type for managing vague structures, Int. J. Intell. Syst. 15 (2000) 1061–1085.
[33] N. Marín, J.M. Medina, O. Pons, D. Sánchez, M.A. Vila, Complex object comparison in a fuzzy context, Inf. Software Technol. 45 (7) (2003) 431–444.
[34] F. Berzal, N. Marín, O. Pons, M.A. Vila, Managing fuzziness on conventional object-oriented platforms, Int. J. Intell. Syst. 22 (7) (2007) 781–803.
[35] L. Cuevas, N. Marín, O. Pons, M.A. Vila, pg4DB: a fuzzy object-relational system, Fuzzy Sets Syst. 159 (12) (2008) 1500–1514.
[36] A. Zvieli, P.P. Chen, Entity-relationship modeling and fuzzy databases, in: Proceedings of the 1986 IEEE International Conference on Data Engineering, 1986, pp. 320–327.
[37] N.A. Chaudhry, J.R. Moyne, E.A. Rundensteiner, An extended database design methodology for uncertain data management, Inf. Sci. 121 (1–2) (1999) 83–112.
[38] E. Ruspini, Imprecision and uncertainty in the entity-relationship model, in: Fuzzy Logic in Knowledge Engineering, Verlag TUV Rheinland, 1986, pp. 18–22.
[39] R.M. Vandenberghe, An extended entity-relationship model for fuzzy databases based on fuzzy truth values, in: Proceedings of the Fourth International Fuzzy Systems Association World Congress, 1991, pp. 280–283.
[40] G. Vert, A. Morris, M. Stock, P. Jankowski, Extending entity-relationship modeling notation to manage fuzzy datasets, in: Proceedings of the Eighth International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, 2000, pp. 1131–1138.
[41] G.Q. Chen, E.E. Kerre, Extending ER/EER concepts towards fuzzy conceptual data modeling, in: Proceedings of the 1998 IEEE International Conference on Fuzzy Systems, vol. 2, 1998, pp. 1320–1325.

[42] Z.M. Ma, W.J. Zhang, W.Y. Ma, G.Q. Chen, Conceptual design of fuzzy object-oriented databases using extended entity-relationship model, Int. J. Intell. Syst. 16 (2001) 697–711.
[43] J. Galindo, A. Urrutia, R.A. Carrasco, M. Piattini, Relaxing constraints in enhanced entity-relationship models using fuzzy quantifiers, IEEE Trans. Fuzzy Syst. 12 (6) (2004) 780–796.
[44] S. Abiteboul, R. Hull, IFO: a formal semantic database model, ACM Trans. Database Syst. 12 (4) (1987) 525–565.
[45] M.A. Vila, J.C. Cubero, J.M. Medina, O. Pons, A conceptual approach for deal with imprecision and uncertainty in object-based data models, Int. J. Intell. Syst. 11 (1996) 791–806.
[46] A. Yazici, B.P. Buckles, F.E. Petry, Handling complex and uncertain information in the ExIFO and NF^2 data models, IEEE Trans. Fuzzy Syst. 7 (6) (1999) 659–676.
[47] Z.M. Ma, A conceptual design methodology for fuzzy relational databases, J. Database Manage. 16 (2) (2005) 66–83.
[48] R. Bouaziz, S. Chakhar, V. Mousseau, S. Ram, A. Telmoudi, Database design and querying within the fuzzy semantic model, Inf. Sci. 177 (21) (2007) 4598–4620.
[49] D. Kucuk, N. Burcuozgur, A. Yazici, M. Koyuncu, A fuzzy conceptual model for multimedia data with a text-based automatic annotation scheme, Int. J. Uncertainty Fuzziness Knowledge-Based Syst. 17 (Supplement) (2009) 135–152.
[50] N.B. Ozgur, M. Koyuncu, A. Yazici, An intelligent fuzzy object-oriented database framework for video database applications, Fuzzy Sets Syst. 160 (15) (2009) 2253–2274.
[51] A. Yazici, Q. Zhu, N. Sun, Semantic data modeling of spatiotemporal database applications, Int. J. Intell. Syst. 16 (7) (2001) 881–904.
[52] T.J. Teorey, D.Q. Yang, J.P. Fry, A logical design methodology for relational databases using the extended entity-relationship model, ACM Comput. Surv. 18 (2) (1986) 197–222.
[53] P. Bosc, H. Prade, An introduction to fuzzy set and possibility theory based approaches to the treatment of uncertainty and imprecision in database management systems, in: Proceedings of the Second Workshop on Uncertainty Management in Information Systems: From Needs to Solutions, 1993.
[54] L.A. Zadeh, Toward a generalized theory of uncertainty (GTU)—an outline, Inf. Sci. 172 (1–2) (2005) 1–40.
[55] L.A. Zadeh, Is there a need for fuzzy logic?, Inf. Sci. 178 (13) (2008) 2751–2779.