BASIC NOTIONS AND RATIONALE OF THE INTEGRATION OF UNCERTAINTY MANAGEMENT AND OBJECT-ORIENTED DATABASES

R. DE CALUWE
Computer Science Laboratory, University of Ghent,
Sint-Pietersnieuwstraat 41, B-9000 Ghent, Belgium

N. VAN GYSEGHEM
Department of Mathematics and Computer Science,
Centenary College of Louisiana,
P.O. Box 41188 Shreveport La 71134-1188, U.S.A.

V. CROSS
Systems Analysis, Miami University,
Oxford, OH 45056, U.S.A.

The field of databases is a very vast field which has been studied by many researchers for many years. Countless books and papers have been published that reflect the increasing importance of this field. The present book contributes to this research by focusing on the integration of the object-oriented database theory with the theory of uncertainty modelling, which is another rapidly developing field. Combining the advantages of both theories permits the design of databases which are semantically far more powerful and user-friendly than the traditional ones. This chapter of the book explains the reasons for this and introduces the basic concepts an terminology of the underlying theories.

1 Data and Databases

The purpose of a database application is to store data, information or knowledge about the application.

The terms "data" and "database", "information" and "knowledge" appear in all chapters of this book, and although they are considered common basic concepts within the field of databases, they deserve some attention here.

- The term data covers a broad scope in meaning, ranging from crude figures and strings, as used for input and output, to their counterparts, which carry specific semantics. For instance, a number can be considered as a mere mathematically defined structure, or can represent an age or a salary.
- A database is a structured collection of data, which are intentionally brought together and made persistent and suited for querying.
- In the context of this book, information means semantically meaningful data which can be derived from a database by querying it in a more or less intelligent way.
- Facts and their logical interrelations are termed knowledge. Knowledge can also be kept in a database, for instance represented in the form of logical rules or via an equivalent formalism. In this case the database is generally called a knowledge base.

From the foregoing, it is clear that a database comprises more than just data and hence, that modelling a database is more complex than modelling data. A database model uses a data model for the description part of its data. The concepts of data modelling and of database modelling are both descriptive, indicating the existence of an underlying structure, in which the static as well as the dynamic aspects are included. The way in which these aspects are addressed is very different between different models.

2 Uncertainty

It is probably impossible to capture all of the semantics of real world information, a fortiori to model it in a perfect way. Frequently, the observation of and the knowledge about the real world are deficient and, as a consequence, its modelling and hence its representation is imperfect in some way.

In written English literature, a deficiency in the knowledge of information is most frequently termed "uncertainty", regardless of the application field [Klir 1995].

Although our world is filled with a wide variety of different kinds of uncertainty, science has typically ignored most uncertainty in its models, and preferred most of the time to consider uncertain information as lacking information. Admitting the use of uncertain information is admitting to take into account the individual interpretation of knowledge. This is not an obvious policy. A very strong tradition in science strives for absolute objectivity and certainty, based on facts and figures of common understanding and by the evaluation and approval of the community, which is so characteristic of our twentieth century lifestyle. At the eve of the third millennium, this tradition seems to be changing, and the handling of uncertainty is being acknowledged as essential to science [Korth 1997]. The interested reader can find such common observations further developed and put in their historical perspective in [Klir 1995], a standard reference work for fuzzy sets and fuzzy logic, and in a very basic French text, [Arago 1994], which also serves as a reference text for fuzzy sets and fuzzy logic.

2.1 The nature of uncertainty

Focusing on the application field of databases, it is of interest to study the different kinds of uncertainty which can exist about the data, about the way to structure data, about the facts involving or relating data and about the meaning of data, taking into account that data eventually can draw their meaning from the comparison with other data through their mutual or supposed proximity or similarity. Therefore, the data model as well as the database model can be affected by the different kinds of uncertainty.

[Motro 1995] gives a classification of uncertainty with respect to database applications, and assumes that uncertainty permeates the models of the real world, but not the real world itself. He uses the term uncertainty to refer to any element of the model that cannot be asserted with complete confidence. He distinguishes:

- uncertainty (when it is not possible to determine whether an assertion in the model is true or false),
- imprecision (when the information available in the model is not as specific as it should be),
- vagueness (when the model includes elements that are inherently vague),
- inconsistency (when the model contains two or more assertions that cannot be true at the same time) and
- ambiguity (when some elements of the model lack complete semantics, leading to several possible interpretations).

The first three kinds of uncertainty in the classification, i.e. uncertainty in its specific meaning, imprecision and vagueness, are explained in more detail here, as they are addressed in the following chapters. The concept of vagueness, which is also called fuzziness, is associated with the inability to define sharp or precise borders for some domain (of information) and therefore represents an inherent uncertainty. The concepts of uncertainty and imprecision cover the deficiency of the information due to a lack of knowledge about the database application. Although there exists a certain affinity among these three concepts, they are orthogonal. They can appear side by side or simultaneously within a database environment.

Some examples should clarify the distinction between the three notions of uncertainty. To start with, consider the concept of "young ages". It is a multi-valued concept, as a number of different ages can be considered as young, and it is a vague (fuzzy) concept, because the borderline between the "young ages" and "not young ages" is not uniquely or well defined. For instance, 1 year, 2 years, 3 years ... are definitely young ages, 30 years and older are definitely not young ages, while 20 years, 21 years, ... are not really young nor really old ages : they are considered young ages to a lesser extent than the ages 1, 2, 3, ...

Technically speaking, a fuzzy concept, such as young ages, has an "and-semantics" : all ages ranging from 1 to 30 years correspond (to some extent) to "young ages".

The former example contrasts with the example of a statement such as "John is between 15 and 25 years old." Every person has a unique and well-defined age, which in this case is not precisely known or is not expressed in a precise way : John's age can be 15 years or 16 years or ... or 25 years old. In this example, the use of "is between 15 and 25 years old" illustrates the concept of imprecision.

Imprecision has an "or-semantics" : among the given ages only one age is the right one, but there is no certainty about which one.

Similarly, a statement such as "the age of John is young" expresses imprecision in the description of John's age. Here, John's age belongs to the vague concept of "young ages", or stated otherwise, John can be 1 year old or 2 years or 3 years or ... or (to a lesser extent) 20 years or 21 years ..., but he definitely is not 30 years or older. Frequently, this kind of imprecise information is also referred to as vague information, because it uses a vague concept to express the imprecision.

Finally, a statement such as "it is highly possible that John is 25 years old" illustrates an uncertainty in a very specific meaning : here it expresses doubt about the information that "John is 25 years old", but not about the use of "25". Such uncertainty can also qualify imprecise information, as in the statements "it is highly possible that John is between 15 and 25 years old" and "it is highly possible that John is young".

2.2 *Modelling uncertainty*

The existence of different kinds of uncertainty results in different ways to model uncertainty, some of which are discussed here.

A substantial part of the discussion is devoted to the fuzzy set theory as a means to model the concept of fuzziness. Another part treats the related possibility theory, as this theory offers a suitable formalism to model imprecision and uncertainty in its specific meaning. The discussion is completed with the notions of similarity and of typicality, which are used to establish degrees of uncertainty.

The fuzzy set and possibility theories have experienced rapid advances over the last decades. Only a limited number of mathematical concepts are discussed here since the contributions of the different authors mainly deal with how uncertainty handling can be part of the powerful capabilities of object-oriented databases, and not per se with the use of the most advanced mathematical novelties in the theories.

2.3 *The fuzzy set theory and related possibility theory*

The existence of a very extensive literature on fuzzy set theory witnesses its ongoing development since its introduction by L. Zadeh in the mid sixties [Zadeh 1965]. Looking for a standard reference text the reader can refer to [Klir 1995].

The concept fuzzy set can be seen as a generalization of the concept crisp set.
A fuzzy set F on a universe U is defined as a set of ordered pairs (x,y), where x is an element of the universe of discourse and y is an element of the closed interval $[0,1]$:

$$F = \{ (x,y) \mid x \in U, 0 \leq y \leq 1 \}$$

The value y is called the membership degree associated with x by the fuzzy set F. A membership degree 1 indicates full membership of x in F, a membership degree 0 indicates no membership in F, and a membership degree $0<y<1$ indicates partial membership of x in F. Partial membership means that the element x belongs to the fuzzy set only up to some degree, which is indicated by the membership degree.

A crisp set can be viewed as a special case of a fuzzy set, for which only the membership degrees 0 and 1 are associated with elements of the universe.

Similar to the characteristic function χ_S of a crisp set S - which is a function $\chi_S : U \rightarrow \{0,1\}$ so that : $\chi_S(x) = 1$ if $x \in S$ and $\chi_S(x) = 0$ if $x \notin S$

for every element x of the universe U - the membership function μ_F of a fuzzy set F is defined as a function $\mu_F : U \rightarrow [0,1]$, which associates with each element $x \in U$ the membership degree y of x in F : $\mu_F(x) = y$.

The actual membership degree of an element in a fuzzy set does not have an absolute meaning. A membership degree $\mu_F(x)$ draws its meaning from a comparison with the membership degrees $\mu_F(u)$ of other elements $u \in U$: if $\mu_F(x) > \mu_F(u)$, i.e. if x has a higher membership degree in F than u, then x belongs to the fuzzy set F with a higher degree than u. If F describes a vague notion, such as the notion of young age, then it is said that the element x resembles this notion better than u.

The support of a fuzzy set F is a subset of the universe U, which contains all elements having a full or partial membership in F : $supp_F = \{x \in U \mid \mu_F(x) > 0\}$

The core of F is a subset of U containing only the elements having a full membership in F :

$$core_F = \{x \in U \mid \mu_F(x) = 1\}$$

The fuzzy set F is normalized if the core is a non-empty set.

The support and the core are two special cases of (strict) α-cuts. An α-cut, respectively a strict α-cut, of a fuzzy set F is a subset of U containing all elements having a membership degree $\geq \alpha$, respectively $> \alpha$, where $\alpha \in [\, 0, 1 \,]$:

$$F_\alpha = \{x \in U \mid \mu_F(x) \geq \alpha\}$$

$$F_{\bar{\alpha}} = \{x \in U \mid \mu_F(x) > \alpha\}$$

Commonly used shapes of the membership function μ_F of a fuzzy set F on the universe of real numbers $\mathbb{R}$ are trapezoidal membership functions. A trapezoidal membership function, shown in Figure 1 is determined by four parameters (a,b,α,β) :

$$x<a-\alpha \ : \ \mu_F(x)=0$$

$$a-\alpha<x<a \ : \ \mu_F(x)=1-\frac{a-x}{\alpha}$$

$$a<x<b \ : \ \mu_F(x)=1$$

$$b<x<b+\beta \ : \ \mu_F(x)=1-\frac{x-b}{\beta}$$

$$b+\alpha<x \ : \ \mu_F(x)=0$$

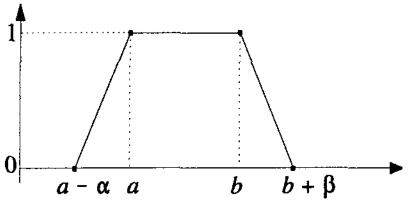

Figure 1 : trapezoidal membership function

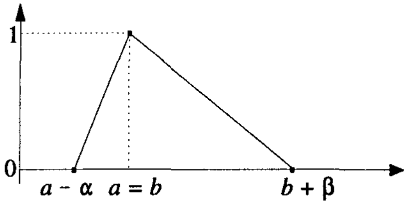

Figure 2 : triangular membership function

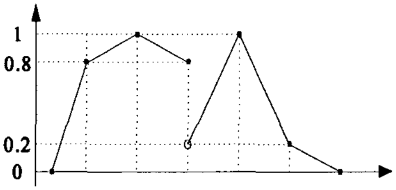

Figure 3 : piecewise linear membership function

If $a=b$, μ_F is a triangular membership function, shown in Figure 2. Another, more general shape for the membership function of a fuzzy set on $\mathbb{R}$ is a piecewise linear membership function, shown in Figure 3. Such membership function μ_F is determined by an ordered list of breaking points. Each breaking point is represented as a quadruple (x_i,l_i,f_i,r_i), where x_i is the real number at which μ_F changes its slope, where f_i is the membership degree of x_i, and where l_i, resp. r_i, are the left limit, resp. right limit of the membership function at x_i:

$$\{(x_1,l_1,f_1,r_1),\ldots,(x_i,l_i,f_i,r_i),\ldots,(x_n,l_n,f_n,r_n)\}$$

The membership degree associated with a real number x in between two consecutive breaking points x_i and x_{i+1} is calculated using linear interpolation :

$$\mu_F(x) = r_i + \frac{(l_{i+1}-r_i)(x-x_i)}{(x_{i+1}-x_i)}$$

Operations defined for crisp sets are extended for fuzzy sets. However, such extensions are not unique. The most stringent condition imposed, is that an extended operation equals the regular operation when it is executed against crisp sets. A general discussion of several definitions for extended operations can be found in [Kli95]. The most frequently used, and most simple extension of set operations was originally proposed by Zadeh :

(regular) inclusion : $F_1 \subseteq F_2 \Leftrightarrow \forall x \in U\colon \mu_{F_1}(x) \leq \mu_{F_2}(x)$

strong inclusion : $F_1 \sqsubseteq F_2 \Leftrightarrow supp_{F_1} \subseteq core_{F_2}$

(regular) equality : $F_1 = F_2 \Leftrightarrow \forall x \in U\colon \mu_{F_1}(x) = \mu_{F_2}(x)$

complement : $\forall x \in U\colon \mu_{co(F)}(x) = 1 - \mu_F(x)$

intersection : $\forall x \in U\colon \mu_{F_1 \cap F_2}(x) = \min(\mu_{F_1}(x), \mu_{F_2}(x))$

union : $\forall x \in U\colon \mu_{F_1 \cup F_2}(x) = \max(\mu_{F_1}(x), \mu_{F_2}(x))$

A fuzzy set of level two is a fuzzy set of fuzzy set, i.e. if $\mathscr{F}(U)$ is the universe of fuzzy sets on U, then a fuzzy set of level two is an element of $\mathscr{F}(\mathscr{F}(U))$.

A fuzzy relationship between the universes U_1, ..., U_n is defined as a fuzzy set on the Cartesian product of the universes $U_1 \times \ldots \times U_n$. A similarity relationship s is a fuzzy binary relationship, defined as a fuzzy set on $U \times U$ which satisfies the properties of reflexivity, symmetry and transitivity :

$$\forall x \in U:\ \mu_s(x,x) = 1$$

$$\forall x,y \in U:\ \mu_s(x,y) = \mu_s(y,x)$$

$$\forall x,y \in U:\ \mu_s(x,y) \geq \max\{\min(\mu_s(x,z),\mu_s(z,y)) \mid \forall z \in U\}$$

The formal definition of a fuzzy measure is beyond the scope of this introductory chapter. What is of interest, is that a fuzzy measure is used to model the uncertainty encountered when determining whether or not an element $x \in U$ is a member of a number of crisp sets. A special fuzzy measure is the possibility measure Π. For any set S on the universe U, $\Pi(S)$ denotes the degree in which S is considered "possible". A possibility measure satisfies the following properties for any set S_1, S_2 and S on U :

$$\Pi(S_1 \cap S_2) = \max\{\Pi(S_1),\Pi(S_2)\}$$

$$\max\{\Pi(S),\Pi(\mathrm{co}(S))\} = 1$$

The last property indicates that at least S or its complement are completely "possible", but, when S is completely possible, it does not prevent its complement from also being completely possible.

For a finite universe U, a possibility measure can be defined by means of a possibility distribution π, which is a function $\pi: U \rightarrow [0,1]$ so that for any set S on U and any element of U :

$$\Pi(S) = \max\{\pi(u) \mid \forall u \in S\}$$

$$\pi(x) = \Pi(\{x\})$$

A set can impose a constraint on the value of a variable A by stating that the value of A must belong to this set. Similarly, a (normalized) fuzzy set F can impose an "elastic" constraint on the value of a variable A. When the membership function μ_F indicates the degrees to which the elements of the universe correspond to a real-world concept, it can be used to restrict the possible values of a variable A :

$$\forall x \in U:\ \pi_A(x) = \mu_F(x)$$

In this case, the membership degree of x in F determines the possibility degree $\pi_A(x)$ in which the variable A equals x. Such elastic constraint is also used to model the possible values of a variable A of which the exact value is not precisely known. For instance, the value of A can be described using a fuzzy concept or vague linguistic term, resulting in the imprecise value. This fuzzy concept is modelled as a fuzzy set, which then determines the possibility distribution π_A modelling the value of A.

The regular operations which operate on crisp, precise values need to be extended to also account for imprecise values. Again, such extension is not unique, but the most frequently used, and most simple extension was originally proposed by Zadeh through the extension principle, which explains how an operation can be extended to fuzzy sets :
if :

$$\begin{aligned} f\colon\ U_1 \times \ldots \times U_n &\to U_{n+1} \\ (x_1,\ldots,x_n) &\to x_{n+1} \end{aligned}$$

then:

$$\begin{aligned} f'\colon\ \mathcal{F}(U_1) \times \ldots \times \mathcal{F}(U_n) &\to \mathcal{F}(U_{n+1}) \\ (X_1,\ldots,X_n) &\to X_{n+1} \end{aligned}$$

where for $1 \le i \le n+1$, X_i are fuzzy sets on U_i, so that :

if :

$$\begin{aligned} f\colon\ U_1 \times \ldots \times U_n &\to U_{n+1} \\ (x_1,\ldots,x_n) &\to x_{n+1} \end{aligned}$$

then :

$$\begin{aligned} f'\colon\ \mathcal{F}(U_1) \times \ldots \times \mathcal{F}(U_n) &\to \mathcal{F}(U_{n+1}) \\ (X_1,\ldots,X_n) &\to X_{n+1} \end{aligned}$$

where for $1 \le i \le n+1$, X_i are fuzzy sets on U_i, so that :

$$\mu_{X_{n+1}}: U_{n+1} \rightarrow [0,1]$$

$$x_{n+1} \rightarrow 0 \qquad \text{if } \nexists\, (x_1,...,x_n) \in U_1\times...\times U_n : x_{n+1}=f(x_1,...,x_n)$$

$$x_{n+1} \rightarrow \max\{\min\{\mu_{X_1}(x_1),...,\mu_{X_n}(x_n)\} \mid \forall\, (x_1,...,x_n) \in U_1\times...\times U_n : x_{n+1}=f(x_1,...,x_n)\}$$

otherwise

The original formulation of this extension principle is given for fuzzy sets without any imposed semantics. When the semantics is taken into account, it is obvious that such extended operation is useful primarily for imprecise values, i.e. for values restricted by an elastic constraint defined by fuzzy sets, or stated otherwise, for values modelled by possibility distributions.

Theories other than the fuzzy set theory and the related possibility theory have been developed to support the handling of inexactness in information. We mention for instance the rough set theory, which can support the modelling of indiscernability or ambiguity; this theory has even been used in combination with the fuzzy set theory for the same purpose [Dubois 1992], [Pawlak 1996]. We also mention the early efforts to explicitly represent the absence of information in a database environment, using the concept of null values and default values, to represent existing standard information. An interesting discussion of some of these more traditional methods is given in [Abiteboul 1995]. For the particular case of object-oriented databases, we refer to [Zicari 1990].

3 Object-Oriented Databases

[Cattell 1992] refers to [Ullman 1989] to state that there exist three kinds of database needs : data management, object management and knowledge management. Only data management is addressed by relational database management systems. Object management is concerned with the requirements for more complex data structures, and is addressed now by object-oriented databases. Knowledge management refers to the broader scope of deriving information about a domain through an inference system; the object-oriented technology is a breeding ground for developing this kind of systems. For instance, integrating objects and rules has already been extensively addressed in [Kim 1990].

The successive development of database models which meet these needs, demonstrates the increasing ability of database systems to master the complexity in database applications.

The older, but still successful, relational database model was first specified in [Codd 1970], the more recent object-oriented database model was first specified in the Object-Oriented Database System Manifesto [Atkinson 1990].

The shift from relational to object-oriented databases went along with the development of other data and database models, each contributing in their own way, but none of them being materialized in fully-fledged commercial products. Indeed, numerous extensions of the relational model paved the way to more powerful functional and semantic models. However, none of the new models has proved to be powerful enough to supplant the relational model as a basis for the leading market products.

Another test for the new models is dictated by economic reasons : for the vendor companies, the return on investment is very important, and for the users, the necessary cost of rewriting applications needs to be as low as possible. Object-oriented technology seems on its way to become successful in meeting this challenge. For sure, the present profusion of scientific and technical books and papers which are devoted to the bridging of the gap between relational and object-oriented database models and products is a good omen. Research on object-oriented and object-relational databases has already been recognized as leading to major achievements in the last five years [Silberschatz 1996]. An in-depth discussion is found in [Kim 1995a].

The difficulty of catching the right moment for the very transition, undoubtedly related to the accomplishments with respect to standardization [Thompson 1995], is once more painfully experienced.

The object-oriented paradigm, dating from the sixties, at first served as a basis for the development of new programming languages. Later on it became the supporting mechanism for much more complex software engineering tools, if not the essence of modern software engineering itself. A historical overview is found in [Graham 1991]. From the database perspective, a seamless integration of programming language and data handling can be achieved, which makes object-oriented database programming less cumbersome and hence, greatly improves the productivity of the programmer.

Object-oriented systems have a significant advantage over their predecessors in that more complex underlying data models can be used and, therefore, a higher level of data abstraction and thus, data independence, is provided. They are suited for more sophisticated applications and more naturally support these kinds of applications in their entirety, as the whole software environment can be built on the same principles, using the same programming tools. Authorization control, transaction management, portability, interoperability, ... all of them can be realized with the same means.

Textbooks and papers which provide the details of using object-oriented systems are abundant. Among the recent reference works on object-oriented databases we wish to mention [Graham 1991] and [Loomis 1995].

A proposal for standardization of object-oriented databases has emerged [Cattell 1994] and was revised shortly after, already including possible further extensions in the same report [Cattell 1996]. The ODMG proposal was sparked by the efforts of the OMG-group, to standardize the object model by defining an architecture in view of creating portable applications, and by the X3/SPARC/DBSSG/OODB Task Group, which describes an object-oriented database as a collection of characteristics. The ODMG group proposal has been established by a consortium of object-oriented database vendors, and is backed by industrial input and academic reviewing, but stands outside of the traditional standard bodies. Nowadays it can be considered as a de facto standard awaiting approval from the standard committees.

3.1 The Object-Oriented Paradigm

Central to the object-oriented paradigm is the notion of an object. Objects stand for real world or abstract entities and consist of encapsulated data structures, possibly of great complexity, having behavioural capacities. Based on their properties of structure and behaviour, objects belong to classes, which can be related to each other in subclass-superclass hierarchies endowed with inheritance. In practice, classifying objects can be a very difficult problem [Booch 1991].

Object-oriented database systems fully exploit the notions of object, class and inheritance; when only the structural part of object-orientation is taken into account, the term object-centered database systems is used. In short, object-oriented databases are sometimes called object databases.

Some of the following chapters study object-centered models, others study additional aspects, up to the treatment of full object-oriented models.

Only a limited number of concepts are essential to all object-oriented modelling. The concepts which are useful for the understanding of the next chapters are introduced hereafter. Unfortunately, sometimes there is a lack of agreement on the terminology.

In the object model, each object has a unique object identifier (OID) which is used to distinguish it from all other objects.

Each object is of a particular type, generally called a class in object-oriented programming languages. The term type is preferred here, since the term class has been used to define both object intent (structure and behaviour) and object extent (set of like objects). The term class has also been loosely used to refer to the implementation of a type since a type can have more than one implementation specification.

Some types known as abstract types are intended as supertypes for other subtypes and may not have objects directly created from them.

Collection types are predefined data organization methods such as sets, bags, lists, and arrays. Objects may be grouped into named instances of one of these collection types.

The extent of an object type is a system-maintained set collection for all the objects of the object type. This extent includes the objects of all subtypes of the type and is also referred to as the maximal extent.

The object type defines the properties, i.e., attributes and relationships, and the operations that make up objects of this type. Each attribute has a type. Attributes are classified as simple attributes or complex attributes. Simple attributes take on literal values. Complex attributes are categorized as reference attributes and collection attributes. Reference attributes hold references to objects, i.e., object identifiers. Collection attributes hold collection object identifiers. Attribute signatures define the attribute name and the type of its legal values.

The other kind of property, a relationship, defines an association between types of objects. In [Cattell 1994] and likewise in [Cattell 1996] only binary relationships are supported. A relationship is also referred to as a reference attribute. The relationship signature specifies the cardinality, i.e., one-to-one, one-to-many or many-to-many, the related object type, and the inverse attribute which refers to an attribute in the related object type.

The inverse attribute is essential to ensure referential integrity. The declaration of an inverse attribute represents the implicit specification of a traversal path. With binary relationships, traversal paths must exist in pairs, one for each direction of traversal of the relationship.

Operations are functions that can be applied on or by objects; they define the behaviour *of objects. An operation is defined in two parts: by its external and by its internal* characteristics, called the interface and its implementation. They are specified as a set of operation signatures and methods. An operation signature defines the name of the operation, the name and type of any parameters, the name and type of any returned values, and the names of any exceptions which the operation might cause to occur. A method for an operation specifies the implementation of the operation. An object's methods are used to encapsulate the semantics of the object.

Object types are linked to other object types in subtype/supertype hierarchies. The main purpose of subtype/supertype hierarchies is to permit generalization which is also referred to as inheritance, subtyping or subclassing. Generalization may be considered as one of the following: specialization, specification, and classification. In specialization, the subtype inherits the properties and operations of the supertype and may choose to override certain properties of the supertype and/or extend its properties when additional properties are required by the subtype. Specification permits a subtype to be defined for a supertype by testing a predicate defined over a set of attributes. Distinguishing between different subtypes might be necessary, for example, if different methods are required for them. In classification, subtypes are simply used as sets to categorize objects. Specification includes classification as a trivial case since a specific attribute could be defined to indicate to which subtype the object belongs. Although these different kinds of generalization exist, database systems and programming languages use the same subtyping mechanism to realize each kind of generalization. This practice may be a reason for some of the confusion existing among object-oriented systems.

Multiple inheritance occurs when a subtype has multiple supertypes. Multiple inheritance is useful when more than one orthogonal classification hierarchy exists.

Database objects which have to outlive the execution of the application programs, are made persistent, in contrast to the other objects which are only transient.

The object model concepts and terminology just reviewed are part of a rapidly changing object technology. The rapid change is being brought about partly by the needs of the applications and by the promises of the database vendors. The object model may serve as a unifying framework for supporting multimedia and complex applications and for integrating with developers' increasingly object-based tools.

4 Uncertainty in Object-Oriented Databases

Uncertainty handling is not frequently addressed in a database environment. This contrasts with other application areas, such as engineering applications, decision problems, control systems, rule based knowledge systems and so on, where uncertainty management has been incorporated and has lead to new and interesting solution methods. These application areas have in common the emphasis of data handling and are much less demanding than database environments which model the information itself.

The majority of the papers concerning fuzzy databases treat formal aspects related to relational databases; practical realizations seem exceptional and either have been developed as prototypes or are the result of private initiatives. Even the most leading references in the literature about uncertainty, such as [Klir 1995], are very brief in their treatment of database applications, at best referring to research about uncertainty in relational databases and not mentioning the efforts of the last decade with respect to object-oriented databases. Moreover, the very few books on fuzziness and databases [Bosc 1995], [Petry 1996] are barely concerned with object-orientation. However, in [Graham 1991], inheritance under uncertainty is briefly considered.

In fact, up to now, only a limited number of research groups have investigated the problem of modelling fuzziness and uncertainty in the context of object-oriented databases. Nevertheless, combined fuzzy and uncertain object-oriented databases offer the power of object-orientation and the availability of a richer semantic expressiveness. The development of theoretical fuzzy and uncertain object-oriented database models is now approaching the point that useful products can begin their initial development.

Of course, as always, research is continuing; at present the definition of a unifying framework for a fuzzy object model is being investigated [Cross 1997].

The next chapters describe proposals for integrating uncertainty in object-oriented database models.

The earlier attempts to endow object-centered models, semantic networks and frame-based models, with fuzzy semantics, are addressed by Rossazza et al. in chapter 2. The proposed model is based on the notion of typicality modelled by fuzzy sets. The model is a static one since all the crisp and fuzzy information is contained in the attributes of the objects themselves. The hierarchical links are based on inclusion definitions related to different implication definitions. As this model is able to handle default knowledge, exceptions can be represented in the model.

In chapter 3 by George et al., the approach of modelling imprecision is based on the similarity concept. The uncertainty in classification and in the inheritance hierarchies in class/subclass relationships, as a consequence of the allowed imprecision in data, is modelled by using fuzzy set theory. An algebra for the defined data model is presented. The practical interest of the theory is illustrated by the development of an extensive example and the discussion of further possible applications.

Extending a graph-based object-oriented data model is presented by Bordogna et al. in chapter 4. The graphical representation goes along with a formal definition of the model, which uses the fuzzy set and possibility theories to represent uncertainty. Uncertainty is investigated here at the schema level as well as at the instance level.

The model of Van Gyseghem et al. is presented in chapter 5. The model is an extension of an object-oriented database model, based on the fuzzy set and possibility theories. It considers uncertainty at the data level as well as the schema level and also discusses the dynamic aspects of object-orientation.

Na et al. present in chapter 6 a fuzzy object data model and a fuzzy association algebra. Fuzzy associations are defined at the metalevel for relationships between classes and at the object level for relationships between objects. Next, patterns of fuzzy associations are defined, and by the algebra, operations on the patterns. The result of the pattern matching operation can be interpreted as the extent to which the retrieved data meet the requirements expressed by the corresponding formulated query.

References

[Abiteboul 1995] Abiteboul, S., Hull, R., and Vianu V., *Foundations of Databases*, Addison-Wesley Publishing Company, Reading, Massachusetts (USA), 1995

[Arago 1994] Observatoire Français des Techniques Avancées (rapport Arago 14) *Logique floue*, Masson, Paris (France), 1994

[Atkinson 1990] Atkinson, M., Bancilhon. F., DeWitt, D., Dittrich, K., Maier, D., Zdonik, S., *The Object-Oriented Database System Manifesto*, in [Kim 1990]

[Booch 1991] Booch, G., *Object-Oriented Design with Applications*, The Benjamin/Cummings Publishing Company, Inc., Redwood City, California (USA), 1991

[Bosc 1995] Bosc, P., and Kacprzyk, J., eds., *Fuzziness in Database Management Systems*, Physica-Verlag, Heidelberg (Germany), 1995

[Cattell 1992] Cattell, R.G.G., *Object Data Management, Object-Oriented and Extended Relational Database Systems*, Addison-Wesley Publishing Company, Reading, Massachusetts (USA), 1991 (reprinted with corrections in 1992)

[Cattell 1994] Cattell, R.G.G., ed., *The Object Database Standard : ODMG-93*, Morgan Kaufmann Publishers Incorporation, San Francisco, California (USA), 1994

[Cattell 1996] Cattell, R.G.G., ed., *The Object Database Standard : ODMG-93, version 1.2*, Morgan Kaufmann Publishers Incorporation, San Francisco, California (USA), 1996

[Codd 1970] Codd, E., *A relational model for large shared data banks*, Communications of the ACM, Vol. 13, No. 6, pp. 377-387, ACM, 1970

[Cross 1997] Cross, V., De Caluwe, R., Van Gyseghem, N., A *Perspective from the Fuzzy Object Data Management Group*, Proceedings of the FUZZ-IEEE'97 Conference, Barcelona, July 1997, IEEE, 1997

[Dubois 1992] Dubois, D., and Prade, H., *Putting Rough Sets and Fuzzy Sets Together*, in [Slowinski 1992]

[Graham 1991] Graham, I., *Object-Oriented Methods*, Addison-Wesley Publishing Company, Wokingham, England, 1991

[Kim 1990] Kim, W., Nicolas J.-M., Nishio S., eds., *Deductive and Object-Oriented Databases*, Elsevier Science Publishers, North-Holland, Amsterdam (the Netherlands), 1990

[Kim 1995a] Kim, W., *Object-Oriented Database Systems : Promises, Reality and Future*, in [Kim 1995b]

[Kim 1995b] Kim, W., ed., *Modern Database Systems, The object model, interoperability and beyond*, Addison-Wesley, Publishing Company,, Reading Massachusetts (USA), 1995

[Klir 1995] Klir, G.J. and Yan, B., *Fuzzy Sets and Fuzzy Logic, Theory and Application*, Prentice Hall P T R, Upper Saddle River, New Jersey (USA), 1995

[Korth 1997] Korth, H.F. and Silberschatz, A., *Database Research Faces the Information Explosion*, Communications of the ACM, Vol. 40, No. 2, pp. 139-142, ACM, 1997

[Loomis 1995] Loomis, M., *Object Databases, The Essentials*, Addison-Wesley Publishing Company, Reading, Massachusetts (USA), 1995

[Motro 1995] Motro, A., *Management of Uncertainty in Database Systems*, in [Kim 1995]

[Pawlak 1996] Pawlak, Z., *Why Rough Sets ?*, Proceedings of the Fifth IEEE, International Conference on Fuzzy Systems, New-Orleans, Sept. 8-11, 1996, Volume 2, pp. 738-743

[Petry 1996] Petry, F., *Fuzzy Databases, Principles and Applications*, Kluwer Academic Publishers, Boston (USA), 1996

[Silberschatz 1996] Siberschatz, A., Stonebraker M. and Ullman J., eds., *Database Research : Achievements and Opportunities into the 21st Century*, SIGMOD Record, Vol. 25, No. 1, March 1996, pp. 52-63, ACM 1996

[Slowinski 1992] Slowinski, R., ed., *Intelligent Decision Support : Handbook of Applications and Advances of Rough Set Theory*, Kluwer Academic Publishers, Boston (USA), 1992

[Thompson 1995] Thompson, C., *The Changing Database Standards Landscape*, in [Kim * 1995]

[Ullman 1989] Ullmann, J., *Principles of Database and Knowledge-Based Systems, Volumes 1 and 2*, Computer Science Press, Rockville, Maryland (USA), 1989

[Zadeh 1965] Zadeh, L., *Fuzzy Sets, Information and Control*, 1965, Volume 8 pp. 338-353

[Zicari 1990] Zicari, R., *Incomplete Information in Object-Oriented Databases*, SIGMOD Record, Vol. 19, No. 3, September 1990, pp. 5-16, ACM 1990