Proposal of Fuzzy Object Oriented Model in Extended JAVA

Wilmer Pereira
Grupo de Investigación en Inteligencia Artificial y Robótica (GIIAR)
Escuela de Ingeniería Informática, Universidad Católica Andrés Bello
Caracas, Venezuela, wpereira@ucab.edu.ve

Abstract. The knowledge imperfections should be considered when modeling complex problems. A solution is to develop a model that reduces the complexity and another option is to represent the imperfections: uncertainty, vagueness and incompleteness in the knowledge base. This paper proposes to extend the classical object oriented architecture in order to allow modeling of problems with intrinsic imperfections. The aim is to use the JAVA object oriented architecture to carry out this objective. In consequence, it is necessary to define the semantics for this extension of JAVA and it will be called Fuzzy JAVA. The NCR `FuzzyJ` library allows represent the vagueness (fuzziness) and uncertainty in class attributes. JAVA extended allows to model fuzzy inheritance.

1 Introduction

In computer science, one of the most popular formalism to model problems, is the object oriented model by their generality and versatility. As the complexity of the problems always increases, it is important to use this methodology in order to make easy the problem representation. However, in many cases, the complexity comes from: the uncertainty, the fuzziness or the incompleteness [10]. Thus, it is desirable to consider theories and methodological tools that represent these knowledge imperfections inside the object oriented model. This would allow the user to specify problems when the knowledge has intrinsic imperfections. In [11] appeared a version of an object oriented model to represent these imperfections using fuzzy sets and possibilistic logic. This proposal was called *Fuzzy Object Oriented Model*. This approach was developed initially for monovalued and multivalued attributes [12]. This paper details a proposal to extend JAVA, considering the knowledge imperfections in: attributes with fuzzy values, fuzzy inheritance and fuzzy or uncertain membership of the objects to the classes.

Precedent papers have revised theses ideas. In [1] is presented an object oriented model of visual data based on graph. It allows to model fuzzy attributes and fuzzy membership from objects to their class. In both cases, it is necessary to use linguistic labels and degrees of uncertainty. With regard to the fuzzy membership of the subclass to the superclass, this paper does not consider it necessary in order to model problems. On the other hand, in [15] is defined the generalized fuzzy sets for fuzzy attributes and fuzzy membership from the objects to class. It defines two degree of uncertainty depending on the attribute membership to his class. There are also two values for the object membership to its class. Here, like in the previous paper, it does not consider the fuzzy inheritance. As for [6], it is defined the membership function for the fuzzy relation: object-class and superclass-subclass. The article has a lot of formulas with not too theoretical support but a good intuition and common sense of the authors. Also in [5] membership functions are specified, characterizing the difference between typical ranges and permissible ranges. The properties between the supports and the kernel of possibility distribution, define the inclusion of superclass-subclass and the membership function from an object to its class. This last article relates quantitative and qualitative aspects that the designer uses to model knowledge base. Our paper defines a semantic for a fuzzy object oriented model on JAVA that we will call *Fuzzy JAVA*. We consult ideas of previous works and we propose new ideas to extend JAVA language structure.

2 Preliminaries

2.1 Fuzzy Sets

These sets are an extension of regular sets because they allow express membership degree of the elements of the universe [7]. These degrees are represented using of a *membership function* whose range is the real interval [0,1]. The membership function of a fuzzy set F is denoted with the symbol μ_F such that: $\mu_F : U \rightarrow [0,1]$. This characteristic function usually has trapezium form and it is defined by a 4-tuple of the universe (*a*,*b*,*c*,*d*).

In logical terms, a trapezium represents a *fuzzy predicate* where the boolean value stem from membership function of the fuzzy set. In this way, for example the fuzzy predicate `youth` is defined on the universe of ages {0 ..125} with the membership function μ_{youth}=(0,0,25,50). The associated trapezium is:

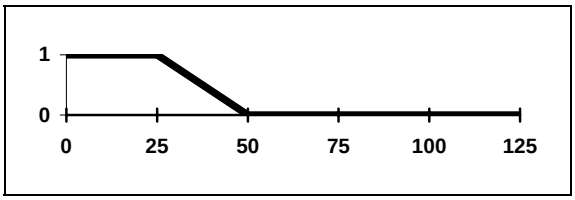

Fig. 1. A fuzzy predicate `youth`

2.2 Possibility Distribution and Possibility Theory

A *possibility distribution* is a function of an universe *U* to the real interval [0,1], just as membership function of the fuzzy sets. A possibility distribution π depend on a fuzzy set *F*: $\pi(x = a) = \mu_F(a)$. With these possibility distributions it is possible to obtain two measures called *possibility* (Π) and *necessity* (N), whose determine the degree of certainty or trust of a statement [4]. This is a less rigorous way to represent the uncertainty than probability theory. The possibility measure is a function whose domain is part of the universe in the interval [0,1], $\Pi:P(U) \rightarrow [0,1]$ such that:

$$\forall A, B \in P(U)\left(\Pi(A \cup B) = max(\Pi(A), \Pi(B))\right) \quad (1)$$

$$\forall A \in P(U)\left(max(\Pi(A), \Pi(\overline{A})) = 1\right) \quad (2)$$

It is important to remark that if *A* and its complement set are totally possible, it represent a situation of total ignorance. This is an advantage of the possibility theory on the probability theory, which does not allow represent ignorance on probabilities. The necessity measure is the dual concept of possibility measure that is also defined on the parts of *U* in the real interval [0,1]. The necessity and possibility measure are weaker than probabilities values because they do not impose conditions on the event probability and its event complement. Besides the sum of the possibility or necessity measures do not have to add 1. Consequently, these measures are better adapted to model of subjective uncertainty according to the appreciations of a designer.

With regard to the operations on fuzzy sets, the intersection is defined like a combination of sets possibilities measures by means of a triangular norm *T*.

$$\mu_{F \cap G}(x) = T\left(\mu_F(x), \mu_G(x)\right) \quad (3)$$

A triangular norm is a binary operator on the real interval [0,1] that is associative, commutative, monotonic, for which 1 is the neutral element. Usually, the intersection uses the biggest triangular norms that is the minimum (MIN). On the contrary, the union of two fuzzy uses a triangular conorm *S*.

$$\mu_{F \cup G}(x) = S\left(\mu_F(x), \mu_G(x)\right) \quad (4)$$

A triangular conorm has duals properties with regard to triangular norm and the union use the maximum (MAX). The properties of the union and the intersection depend on the measure triangular pair.

Finally another operator used in this paper, is the *α-cut*. This operator links fuzzy sets with traditional sets. This traditional set contains all the elements whose membership to the fuzzy set is bigger than a minimum level of tolerance. The *α*-cut of a fuzzy set *F* is denoted by F_α

$$F_\alpha = \left\{x \in U / \mu_F(x) \geq \alpha\right\} \quad (5)$$

2.3 NRC Fuzzy Library for JAVA

This JAVA library defines and manipulates fuzzy concepts in order to obtain fuzzy reasonings [9]. This tool was created by the NRC (National Research Council of Canada). `FuzzyJ` is focused fundamentally in the fuzzy concepts, by means of the creation and manipulation of membership functions. The most important classes in this library or API are:

`FuzzyVariable` class: An object of this class specifies the universe that will be used to manipulate a fuzzy concept, for example temperature, pressure, etc. To create an instance of this class is necessary to indicate as parameters: a name, the superior and inferior limit of the universe and the variable units. For example:

```
FuzzyVariable temp = new FuzzyVariable(“temperature”, 0, 100, “C”);
```

`FuzzySet` class: The objects of this class allow define possibility functions. To create an instance of this class, is necessary to indicate as parameters an array of `x` values, an array of `y` values (which define the `FuzzySet`) and also the cardinality of the arrays. For example:

```
double xvalues [] = {0.1, 0.3, 0.4, 0.5, 0.8};
double yvalues [] = {0.0, 1.0, 0.65, 1.0, 0.0};
FuzzySet example = new FuzzySet(xvalues, yvalues, 5);
```

Then, it is necessary to add possibility functions on a predetermined universe. The method `addTerm` of the class `FuzzyVariable`, receives as parameters a name and the possibility function.

```
temp.addTerm(“Cold”, new TrapezoidFuzzySet(0.0,0.0,5.0,15.0));
```

`FuzzyValue` class: An object of this class is used to define a group of values `x` and `y`. These values determine the form of a possibility function. There is not a specific concept associated to this function. The `FuzzyValue` associates a `FuzzySet` to a `FuzzyVariable`, that is to say, it associates the possibility function to a universe. The `FuzzyValue` generally, but not always, provides a linguistic expression that gives a comprehensible meaning to the `FuzzySet`.

3 Fuzzy Object Oriented Model

This paper presents a study of a possible fuzzy semantics for: the attributes of objects, the fuzzy inheritance and the fuzzy membership of objects to classes.

3.1 Fuzzy Attributes

There are several ways to associate a fuzzy characteristic to an attribute. There are two classical dimensions on attributes values: the vagueness (fuzziness) with possibility distribution and the uncertainty with possibilistic value. In [12] besides the attributes was presented as monovalued and multivalued but in this paper is presented only fuzzy monovalued attributes.

3.1.1 Monovalued Attributes

A monovalued attribute takes a value which defines its state. For example, a class `person` can contain as attributes: `IDNumber`, `Age`, `Height`, `Address`, `EducationDegree`, `TelephoneNumber`, `PreferedColors`, `FirstLanguage` and `SecondLanguage`. Each individual `person`, for example, `Nathalie` is an object, that is to say, an instance of `person` class. The object `Nathalie` will have specific values for each one of these attributes. For example, the attribute `Age = 8` or `TelephoneNumber = 2426290`. Methods carry out operations on person objects changing their state. For example, `ChangeAddress`, `NewDegree`, etc. In the classical objects oriented model is assumed that all the attributes take a precise value. However we will consider imperfections like fuzziness and uncertainty.

Fuzzines:
An attribute can take a linguistic label associated to a possibilistic distribution, in other words, a vague or fuzzy value [12]. In this case, a `person` object can have assigned the linguistic label `youth`. This defines a possibility distribution whose domain is the possible ages and the range is on [0,1], μ_{youth} : `[1,125]` $\rightarrow$ `[0,1]` (see Figure 1). So if the age of `Nathalie` object is `youth`, then the possibility distribution associated to the attribute `Age` of the object `Nathalie` is:

```
{1/1,1/2,...,1/25,0.95/26,0.9/27,...,0.1/48,0.05/49,0/50,...,0/125}
```

In NCR `FuzzyJ` it is represented in this way:

```
FuzzyValue age = null; …

FuzzyVariable uddAge = new FuzzyVariable(“Age”, 0.0, 100.0, “years”);

uddAge.addTerm(“Youth”, new RightLinearFuzzySet(40.0, 60.0));

nathalie.age = new FuzzyValue(uddAge, “youth”);
```

Uncertainty:
Let us suppose a monovalued attribute with a precise value but who defines it (designer), assigns it a degree of uncertainty, which can turns like a possibility measure. For example, let us suppose the object `Wilmer` that has the value of attribute `SecondLanguage` in English with a degree of uncertainty or possibility of 0.6. The representation in JAVA is:

```
(unc Uncertainty) Attribute = Value;
```

where *Attribute* is a variable of a primitive type or an object `String`, *Value* is an element of the attribute type and *Uncertainty* is a real value, `double` type, belonging to [0,1] interval (possibility measure). The default possibility value is 1. The representation for previous example is

```
(unc 0.6) wilmer.SecondLanguage = English;
```

To recover the uncertainty of an attribute, it is made calling the method:

```
unc(Attribute)
```

Fuzziness+Uncertainty:

A monovalued attribute with these two kinds of imperfections, can be defined like a λ-trapezium [2]. For example let us suppose an object `Nathalie` with its `Height` attribute whose value is the linguistic label: "`more or less 1m10`". The designer adds 0.3 possible of uncertainty to this affirmation. In other terms the designer wants to model: "*it is 0.3 possible that the height of Nathalie is of more or less 1m10*". The trapezium that represents `more or less 1m10` is:

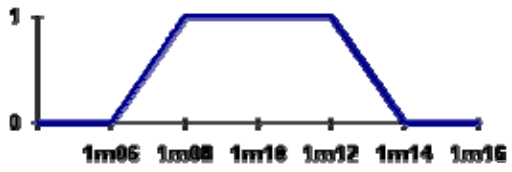

Fig. 2: Trapezium of the fuzzy label more or less 1m10

The possibility of 0.3 on the attribute `height` of the object `Nathalie` introduces a λ-support on the λ-trapezium. When the possibility measure is 1, it represents total uncertainty.

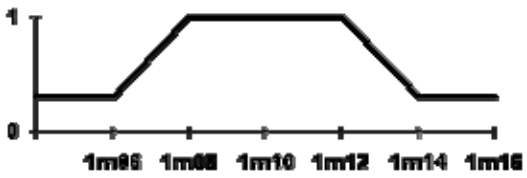

Fig. 3: λ-trapezium of the fuzzy label more or less 1m10

To model this expression implies to assign to an attribute a fuzzy value and a degree of uncertainty. In this way an object `FuzzyValue` should be created, with the possibility measure:

```
(iunc Uncertainty) Variable FuzzyValue = Value;
```

It is important to remark that, in this case, `iunc` is used instead of `unc`. The cause is that default value for `unc`, only for a precise attribute, is 1. While the default value for `iunc`, always on an imprecise or fuzzy attribute, is 0. This is evident for the way to calculate the λ-trapezium. An example in `FuzzyJ` of this mixture of imperfections in the knowledge (fuzziness + uncertainty) is:

```
FuzzyValue height = null;

FuzzyVariable uddHeight=new FuzzyVariable(“height”, 0.0, 230.0, “cm”);

uddEstatura.addTerm(“moreorless110”,new

     TrapezoidFuzzySet(106.0,108.0,112.0,114.0));

(iunc 0.3) height = new FuzzyValue(uddHeight, “moreorless110”);
```

3.1.2 Multivalued Attributes

The situation is different with the multivalued attributes. It represents properties of an object whose values are not atomic but a group of values. For example the attribute `TelephoneNumber` of the object `Wilmer` would be all the telephone numbers where that person can be contacted. Besides it could still be a more flexible mechanism that allows that the values of a multivalued attribute instead of being a classic set would be a fuzzy set [12]. For example the `PreferedColor` attribute belonging to the class `person` can be a fuzzy multivalued attribute. The object `Wilmer` could define its group of prefered colors but also to put them a degree of uncertainty that expresses the level of preference for some color. You can find more precisions in [12].

3.2 Fuzzy Inheritance

In the classical object oriented model, a class inherits all properties of its superclass, as much methods as attributes (except for overwrite rules). In the fuzzy object oriented model, it can be uncertain the application of the properties of a superclass toward the subclass. The cause is, the designer can create a class that inherits of another with a certain level of uncertainty. This will be expressed this way:

```
class subclass extends superclass (imem membership) {…}
```

For example:

```
class coniferous extends tree (imem 0.78) {…}
```

In the multiple inheritance, like in the simple inheritance, it is possible the fuzzy membership a class to their superclass. In other words, a subclass can have a degree of partial inclusion toward different superclass. Since in classic JAVA does not exist

the multiple inheritance, the way to represent it in fuzzy JAVA it is by means of the interfaces:

```
class subclass implements interfaz (imem membership) {,interfaz (imem membership)} {…}
```

For example:

```
class roundedsquare implements circle (imem 0.8), square (imem 0.6) {…}
```

As much for the simple inheritance as for the multiple inheritance, when membership is not specified from a class to its superclass, this it will be assumed that the degree of inclusion of the superclass in the subclass is 1. For examples more specific, see [13].

3.3 Fuzzy Membership of an Object to their Class

In fuzzy object oriented model, an object can have a degree of membership to its class. This degree can be specified when the object is declared or it can be calculated automatically. The explicit definition of object possibility measure on the class, is expressed to the moment to create the object.

```
NomClass NomObject = new NomClass(Parameters) (omem membership);
```

For example:

```
Dinosaur dino = new Dinosaur () (omem 0.5);
```

With regard to the level of membership calculated automatically from the object to the class, an example of multiple inheritance is presented to illustrate better this concepts. Let us suppose a hierarchy of dinosaurs where is obvious the fuzzy character of the classification for the difficulty of identify without ambiguity the evidences. The specialists in dinosaurs only have incomplete fossils to determine the characteristics of a dinosaur and to classify it in the well-known hierarchy until that moment. In this hierarchy of dinosaurs classes the relationship or inheritance is fuzzy, the attributes are not necessary and the membership of each dinosaur to the classification is uncertain.

It is clear that the uncertainty expands for the classification due to the fuzzy and imprecise character of the objects, the inheritance and the attributes. Next the necessary calculations are presented (relative to the norm and triangular conorm) to calculate like it extends the uncertainty and the fuzziness.

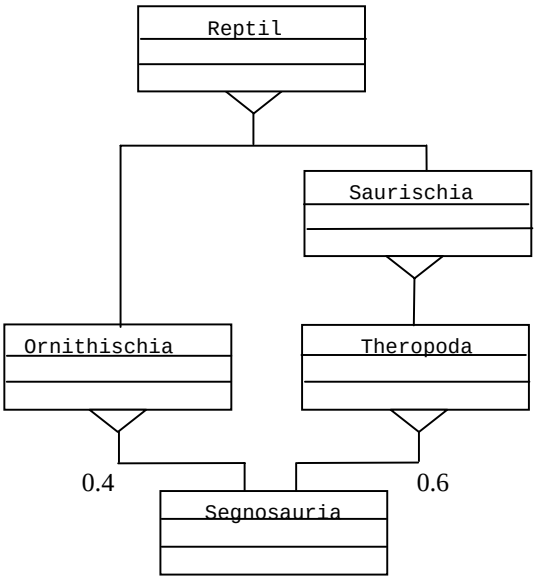

Fig. 4: A fuzzy portion of the hierarchical classification of the dinosaurs

Let us suppose a particular object of the `Segnosauria` class and it will be called `sigy`. Since the anthropologist has not enough documentation on the segnosaurios, `sigy` is believed with a measure of possibility of 0.5:

```
Segnosauria sigy = new Segnosauria () (omem 0.5);
```

Now which the measure of `sigy` possibility is with regard to the superclass `Ornithischia` and `Theropoda`?. It is reasonable that the possibility measure of the object with regard to an immediate superclass is the norm triangular "minimum" (formula 3). In this case the minimum would be between the possibility measure of the object and the possibility measure of the inheritance relationship. The calculations are carried out lineally, as it ascends in the hierarchy toward the superior levels. This way, `sigy` will have as measures of possibility 0.4 with regard to the superclass `Ornithischia` and 0.5 towards the superclass `Theropoda`. When an inheritance relationship does not have an explicit value, it has as possibility measure 1. Therefore, with regard to the superclass `Saurischia`, `sigy` has a possibility measure of 0.5 because the norm triangular minimum, that is to say, min{1, 0.5}.

Now let us see that it happens to `sigy` with regard to the `Reptilian` superclass. There are two "path" in the inheritance hierarchy that converge the superclass, Which the `sigy` possibility degree is in this case? Here it is reasonable to calculate the possibility measure like the conorm triangular "maximum" for the values that converge in the superclass (formula 4). In this case `sigy` has as possibility degree 0.5 to because it is the maximum of the possibilities degrees that converge in the

`reptilian` superclass, max{0.4, 0.5}. Recapitulating, the measures of `sigy` possibility are:

0.4 for the `ornithischia` class
0.5 for the `theropoda` class
0.5 for the `saurischia` class
0.5 for the `reptilian` class

In conclusion, when the object moves lineally in the hierarchy of fuzzy inheritance, the calculation is carried out like a conjunction, that is to say, by means of the bigger than the triangular norms: the minimum. On the other hand, when the object is in an intersection, the calculation is made like a disjunction, that is to say, with the smaller than the triangular conorms: the maximum.

3.4 Selective Inheritance of Methods and Attributes

In classical object oriented model, it is clear that the objects of a class have all the properties from the class to which it belong. However, in the fuzzy object oriented model, because the membership of an object to its class can be uncertain, the applicability of the properties of the class to the object also is uncertain. For this reason, it is defined thresholds like a minimum level of tolerance that the programmer assigns in time of compilation. This allows determine when the objects of the class can use method or attributes, taking into account the level of membership from these to the class. This is represented with α-cut. By default the methods that do not have threshold will only be able to be applied by objects with membership degree 1. This way it is avoided to apply methods that would be insecure for certain objects. The form of assigning a threshold to a method is the following one:

```
(thr threshold) TypeReturn NomMethod (Parameters) {…}
```

For example, the class `Adult` has the methods `puberty` that returns a boolean and `MarriedYears` returns an integer, each one with thresholds 0.3 and 0.7 respectively:

```
(thr 0.3) boolean puberty () {return (age <18);}
(thr 0.7) int MarriedYears(int years) {return (2004 - years);}
```

The object `María` of the class `Adult` is believed with a membership degree of 0.4. On this object one will be able to apply the method `puberty` because it possesses a smaller threshold to the membership degree of `María`, but one will not be able to apply `MarriedYears` because the threshold of the method is higher than the membership degree of the object. In form similar to the methods, an attribute of a class is applicable to an object according to its membership degree. The attributes threshold is defined by the programmer in time of compilation. The form of assigning a threshold to an attribute is:

```
(thr Threshold) Type Attribute;
```

4 Conclusions

The present work defines the semantics of a fuzzy object oriented model using JAVA object oriented model. Tough the API `FuzzyJ` adds fuzzy concepts in applications, it is not comparable with this our approach because we modify the own object oriented model of JAVA. This API only represents and manipulates fuzzy information using a traditional object oriented model.

At this moment we have a prototype for fuzzy JAVA only translates in JAVA classic for the monovalued and multivalued attributes [3]. The following step is to define the grammar that allows us to have an automatic translator from fuzzy JAVA to JAVA classic.

5 Bibliography

[1] G. Bordogna, D. Lucarella and G. Pasi., A Fuzzy Object Oriented Data Model, *III IEEE Int. Conference on Fuzzy Systems*, 1994.

[2] J. C. Buisson, H Farreny and H. Prade, Dealing with Imprecisión and Uncertainty in Expert System DIABETO-III, *Innov. Tech. Biol. Med.,* Vol. 8, N. 2, 1987.

[3] J. Costa and M. Mijares, Análisis Diseño e Implementación de un Modelo Orientado a Objetos Difuso (OOD), Tesis de Grado de la Escuela de Ingeniería Informática, UCAB, 2004

[4] D. Dubois and H. Prade, Théorie des Possibilités: Applications a la Représentation des Connaissances en Informatique, *Editorial Masson*, 1988.

[5] D. Dubois, H. Prade and J-P. Rossazza, Vagueness, Typicality, and Uncertainty in Class Hierarchies *International Journal of Intelligente Systems*, Vol. 6, 1991.

[6] R. George, B.P. Buckles and F.E.Petry, Modelling Class Hierarchies in the Fuzzy Object-Oriented Data Model, *Fuzzy Sets and Systems*, Vol. 60, 1993.

[7] J.P. Haton et al, Raisonnement en Intelligence Artificielle, *InterEditions*, Paris, 1991.

[8] Y. Inoue, S. Yamamoto and S. Yasunobu, Fuzzy Set Object: Fuzzy Set as First-Class Object, *IFSA '91 Conference Proceedings*, Brussels, 1991.

[9] R. Orchard, NCR FuzzyJ Toolkit for the Java Platform; User's Guide, (`http://ai.iit.nrc.ca/IR_public/fuzzy/fuzzyJToolkit2.html`) *National Research Council Canada*, 2003.

[10] W. Pereira, Une Logique Modale pour le Raisonnement dans l'Incertain, *Tesis de Doctorado, Universidad de Rennes I*, Francia, 1992.

[11] W. Pereira and L. Tineo, Modelo Orientado a Objetos Difuso, *L Convención Anual ASOVAC*, Universidad Simón Bolívar, Noviembre/2000, Caracas, Venezuela.

[12] W. Pereira, Atributos Monovaluados y Multivaluados sobre un Modelo Orientado a Objetos Difuso, *1eras Jornadas de Investigación de la UCAB*, 27-28 Noviembre 2001, Caracas, Venezuela.

[13] W. Pereira, Imprecisión e Incertidumbre en un Modelo Orientado a Objetos Difuso, *Trabajo de Ascenso*, UCAB, 2002, Caracas, Venezuela.

[14] A. Thayse et al, Approche Logique de l´Intelligence Artficielle, Volume I y II, *Dunod Informatique*, Paris, 1989.

[15] N. Van Gyseghem, R. De Caluwe and R. Vandenberghe, UFO: Uncertainty and Fuzziness in Object-oriented model, *II IEEE Int. Conference on Fuzzy Systems*, San Francisco, 1993.

[16] S. Yamamoto, Y. Inoue, and S. Yasunobu, Object-Oriented Approaches for Fuzzy Information Processing, *IFES91'*, 1991.