경영정보학연구
제17권 제1호
2007년 3월

확장개체모델에서의 학습과 계층파악

김 용 재*, 안 준 모**, 이 석 준*

Learning and Classification in the Extensional Object Model

Yong Jae Kim, Joon M. An, Seokjun Lee

Quiet often, an organization tries to grapple with inconsistent and partial information to generate relevant information to support decision making and action. As such, an organization scans the environment, interprets scanned data, executes actions, and learns from feedback of actions, which boils down to computational interpretations and learning in terms of machine learning, statistics, and database. The ExOM proposed in this paper is geared to facilitate such knowledge discovery found in large databases in a most flexible manner. It supports a broad range of learning and classification styles and integrates them with traditional database functions. The learning and classification components of the ExOM are tightly integrated so that learning and classification of objects is less burdensome to ordinary users. A brief sketch of a strategy as to the expressiveness of terminological language is followed by a description of prototype implementation of the learning and classification components of the ExOM.

Keywords : Object-oriented Databases, Machine Learning, Query Language, Inheritance, Classification

* Authors are affiliated with College of Business Administration, Konkuk University
** Corresponding author, Joon M. An, College of Business Administration, Konkuk University

I. Introduction

Organizational computing has received a great deal of attention from both computer scientists and organizational researchers because of the increasing strategic importance of information technology in an organization's success. The nature of organizational tasks is to cope with conflicting, inconsistent, and partial information to generate sound, relevant, and reliable information to support decision making and action. It is Daft and Weick[1984] that proposed a model of organizations as interpretation systems to meet the demands of organizational tasks: an organization scans the environment, interprets scanned data, executes actions, and learns from the feedback of actions. Organizational learning is the process by which knowledge of action-outcome relationships between the organization and its environment is obtained.

From a computational perspective, work in machine learning, statistics, active databases, and data visualization addresses capabilities envisioned in Daft and Weick's model. As an integration of these research areas, Matheus, Chan, and Piatetsky-Shapiro[1993] proposed systems for knowledge discovery in databases (KDD) as a collection of components to identify interesting and useful patterns from large databases. A KDD controls the discovery process, interfaces with a database and knowledge base, determines what part of a database in which to focus, extracts interesting patterns, and evaluates extracted patterns. In a similar context, Rao and An[1995] proposed that problem solving tools in group decision making would influence decision scheme, information search, and perceived complexity.

However KDD has not been widely accepted in business and research communities partly for lack of tight integration of knowledge with databases and environments conducive to organizational learning[Goebel and Gruenwald 1999]. In this paper, we propose the Extensional Object Model(ExOM) as a formalism for KDD systems. Classification in the ExOM means testing an object for membership in a given category by applying known rules. Classification involves searching rules in an efficient manner and matching a rule to the attribute values of an object. Learning in the ExOM means generating membership rules and possibly determining groupings for a set of objects. This definition of learning covers supervised learning tasks where the categories are known but the concept definitions defining the categories are not pinned down. Presumably an expert provides a training set with examples from each category of interest. Learning in the ExOM also covers unsupervised learning where neither the grouping of objects into categories nor the membership rules are known. If the groupings of customers are of interest (say potentially high-risk customers), membership rules can be generated to discriminate the different categories. Another kind of unsupervised learning task involves discovery and explanation of deviations from reference values. For example, an important deviation might be customers with large loan balances who change from a high to low quick ratio[1].

1) The quick ratio is the ratio of current assets to current liabilities. It is a measure of a firm's ability to meet short term debt obligations.

The learning task is to discover that a significant subset of customers have this condition and to define a rule that reliably predicts this subset of customers.

The ExOM contributes to the work on KDDs by supporting a wide range of learning and classification styles. One of the key elements underlying the versatility of the ExOM is the concept definition, which is a structure generated by a learning algorithm and interpreted by a classifier. The ExOM supports multiple kinds of concept definitions and permits new kinds of concept definitions to be easily added. The prototype implementation supports four kinds of concept definitions: record expressions, weighted feature records, decision trees, and if-then rules. For each kind of concept definition, multiple learning algorithms can be attached. Learning algorithms are parameterized by the dimensions of example acquisition (incremental or batch), category space (flat or hierarchical), supervision level (supervised, partially supervised, or unsupervised), domain knowledge (relevant attributes for identifying categories and forming concept definitions) and category constraints (disjoint and covering). Multiple classification functions can also be defined for each concept definition so that a user can flexibly select most suitable classification functions.

The ExOM is tightly integrated with an object-oriented data model so that the results of learning and classification can be easily applied in the design and operation of databases. While traditional data models leave learning and classifying of objects as a direct burden to the user, the ExOM supports tight links between the category set and its associated learning and classification algorithms. A category set is a collection of categories that are learned and classified together. Closely associated with category sets are operators that support data analysis and usage of learned results during database operation. The data analysis operators specify a set of categories for learning and classification, determine a set of objects from which to apply a learning algorithm, and conduct simple learning experiments. The results of a learning algorithm can be used for object membership test in a category, iterate over members of virtual category sets, and explain membership of an object in a category.

The rest of this paper is organized as follows. Section 2 provides an informal introduction of the ExOM through an extended example. Section 3 presents a detailed description of the learning and classification components including formal definitions underlying ExOM databases. Section 4 presents properties of ExOM databases including a subsumption testing rule and strategies to use this rule with a taxonomic reasoner. Section 5 depicts a prototype implementation of the ExOM in Smalltalk. Section 6 discusses related work in object-oriented databases, machine learning, and non-traditional database systems. Section 7 summarizes and discusses future directions.

Ⅱ. Overview of the ExOM

We provide an informal overview of the ExOM database through an example of a marketing promotion database, basing our development on the DealMaker system described in chapters 10 and 11 of McCann and Gallagher

[1990][2]. The consumer goods industry has been shifting its focus from product brands to retailers, more emphasizing trade related programs such as cooperative advertising allowances and price discounts. The marketing responsibility for trade related programs is typically shared between a centralized brand group and decentralized sales management and personnel. A fundamental operation is the design and evaluation of a deal which is a set of promotions offered to a retailer in a specific market for a particular brand item. The ExOM is to help the brand manager needs to know whether a deal is likely to be accepted and if yes whether it is likely to be profitable.

2.1 Schema Definition

The fundamental part on an ExOM schema is the representation of categories and their relationships. We use the term category instead of class to emphasize the broader role of classification and learning in the ExOM than traditional object-oriented models. The ExOM features two kinds of categories: (i) structured categories, a kind of deductive representation and (ii) unstructured categories, a kind of inductive representation. Structured categories are similar to classes in traditional object-oriented databases in that the category definition is directly provided by a designer. In addition, the category definition typically provides a necessary condition on membership. In this sense, the category definition serves as an integrity constraint. We first provide a discussion and examples of structured categories followed by unstructured categories. Figure 1 shows prominent structured categories of the promotion database. In the notation, the list of attributes appears in square brackets. The numbers in parentheses following an attribute's type name are the minimum and maximum cardinality, respectively. User defined types (Percent, Money, Container, and City) are assumed to be previously defined. A deal represents a marketing promotion offered to a retailer in a market for a specific item. The terms of a deal can include a price discount and cooperative advertising allowance as well as requirements to order a minimum quantity and advertise for at least a specified time period. As a subcategory of Item, FoodItem inherits its attributes. Retailers stock items, conduct business in a number of markets, and participate in deals. A market is a geographic region in which the leading brands and the sensitivity of features, displays, and prices are monitored. Item statistics such as market share and average price are recorded per market and time interval.

As mentioned previously, the type expression of a structured category (i.e., the attributes and restrictions) typically provides a necessary condition of membership. In this sense, the type expression defines an assertion stating that membership in the category implies the type expression. In the spirit of deductive object-oriented databases, the ExOM also permits the type expression to stipulate a necessary and suffi-

2) Our emphasis will be on the learning and classification aspects of the ExOM in a rudimentary fashion. A more precise and complete description of the ExOM and the syntax description could be secured via contacting authors.

cient condition of membership. The keywords Defined As signify that the type expression defines necessary and sufficient conditions of membership. By using type expressions as assertions and definitions, powerful deductive reasoning can be performed in which a new category definition can be checked for consistency against known category definitions.

In contrast to structured categories where a designer provides the category definition, a learning algorithm determines definitions of unstructured categories. Membership in unstructured categories is based on a concept definition, a structure generated by a learning algorithm and used by a classification function. Given a set of examples about a collection of categories, a learning algorithm generates concept definitions to discriminate among the categories. For instances whose category membership is unknown, a classifier determines membership in a collection of categories given the concept definitions generated by the learning algorithm. Thus, a concept definition is closely associated with a learning algorithm and classification function. The expressiveness of concept definitions is governed by the ability of learning algorithms to efficiently search the space of underlying concepts. In contrast, the expressiveness of type expressions for structured categories is determined by deductive reasoning.

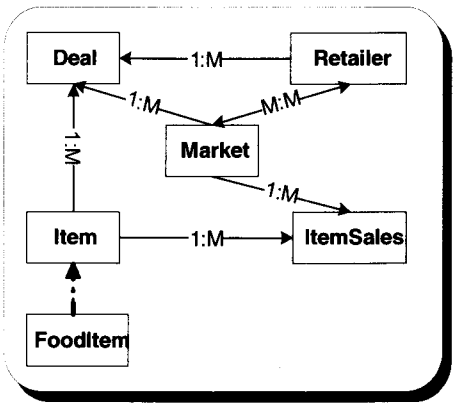
In addition to type expressions and concept definitions, triggers can be used for integrity constraints as defined in Ceri and Widom[1990]. As an example of a trigger, consider the following example that notifies the manager of a deal if the discount is greater than 50%.

```
TRIGGER LargeDiscount In Deal
    On Insert
     If discount > 0.5
        Then notify(manager.name)
```

In continuing with our promotion database example, we define two unstructured categories including their method of membership testing. These two categories serve as predictors of deals likely to be accepted and rejected, respectively. The membership conditions of AcceptedDeal contain two weighted feature records where a weight is enclosed in parentheses following an attribute-value pair. The first specifies a deal involving a certain brand, store, location, and advertising allowance, while the second involves only the store and a discount. The WeightedFeature records were specified by the database designer. A learning algorithm cannot be used with an individual category because learning algorithms need examples of multiple classes. Membership testing is performed using 'WeightedFeatureTest' as specified.

```
UNSTRUCTURED CATEGORY  AcceptedDeal
  Parents  Deal
  Membership Condition
    [item.brand.name = 'topbrand1' (1.0),
       store.name = 'Chain1' (1.0),
       store.location ='Denver' (0.8),
       coop-advertise = 0.2 (0.75)]
    [store.name = 'Chain2' (1.0),
        discount = 0.3 (0.8)]
  Classification By WeightedFeatureTest

UNSTRUCTURED CATEGORY  RejectedDeal
  Parents  Deal
  Membership Condition
```



STRUCTURED CATEGORY Deal
[dealno :: Integer, item:: Item(1,1), store :: Retailer(0,1), mkt:: Market(1,n),
discount :: Percent(0,1), coop-advertise :: Percent(0,1), min-order :: Integer(0,1),
min-ad-time :: Integer(0,1), manager :: Employee(0,1)] Key dealno

STRUCTURED CATEGORY Item
[upc:: String, brand :: String(1,1), price :: Money(0,1), weight :: Float(0,1),
deals :: Set of Deal(0,n), retailers :: Set of Retailer (0,n)] Key upc

STRUCTURED CATEGORY FoodItem
Parents Item [life :: Float(1,1), storage :: Container(1,1)]

STRUCTURED CATEGORY Retailer
[retailno :: String(1,1), name :: String(1,1), markets :: Set of Market(1,n),
items :: Set of Items(0,n), margin :: Percent(0,1), deals :: Set of Deal (0,n)] Key retailno

STRUCTURED CATEGORY Market
[name :: String(1,1), top-brands :: Set of String(0,n), price-sensitivity :: Float(0,1),
feature-sensitivity :: Float(0,1), display-sensitivity :: Float(1,1),
retailers :: Set of Retailer(0,n)] Key name

STRUCTURED CATEGORY ItemSales
[item :: Item(1,1), mkt :: Market(1,1), period :: TimeInterval(1,1),
vol-sales :: Float(1,1), dollar-sales :: Money(1,1), vol-share :: Percent(0,1),
dollar-share :: Percent(0,1), avg-reg-price :: Money(1,1), avg-reduction :: Percent(0,1),
retailers :: Set of Retailer (0,n)] Key item, mkt, period

<Figure 1> Schema of the Market Promotion Database>

[store.name = 'Chain3' (1.0),
store.location = 'Austin' (0.8),
minorder = 20 (0.9)]
Classification By WeightedFeatureTest

In order to use a learning algorithm, categories are grouped into category sets. A category set is defined by specifying the member categories (a tree of categories), learning algorithm, and

classification function. For compatibility, the learning algorithm and classification function must use the same kind of concept definition. Definition of a category set depends on the kind of learning algorithm. If the learning algorithm is supervised, the member categories must be defined before the learning algorithm is executed. If the learning algorithm is unsupervised, the member categories are determined by the learning algorithm and not specified directly.

The following example revises the above concept definitions. The category set DealStatus includes the categories AcceptedDeal and RejectedDeal, using the stated functions for learning and classifying weighted feature records. The concept definition is not given here because the category set DealStatus is generated by the learning algorithm. For purposes here, we assume that the learning algorithm generates the same weighted feature records as in the previous example.

```
CATEGORY SET DealStatus
  Root Deal
  Contains [AcceptedDeal, RejectedDeal]
  Classification By WeightedFeatureTest
  Learning Through InstanceAlgorithm1
```

The ExOM is very flexible with regards to learning and classification. The database designer can define new learning algorithms and classification functions as well as new concept definitions. The latter is equivalent to defining a new class of learning algorithms. Properties of classifiers and learning algorithms are discussed in Sections 3.2 and 3.3, respectively.

2.2 Queries and Operators

The ExOM supports conventional queries as well as operators for learning and classification. Queries consist of a qualification with the usual relational operators and Boolean connectives as demonstrated in the following examples.

```
RETRIEVE All From Deal Satisfies
    item.brand = 'Brand1' AND discount > 0.3
RETRIEVE name From Retailer Satisfies
    location = 'Denver' AND
    items-carried Satisfies-Some (weight > 10
    AND storage = 'Freezer')
```

The second query above demonstrates implicit membership testing: if an Item instance is not a FoodItem, it will not satisfy the condition on the storage attribute. Explicit membership testing can be performed using the In or Member comparison operators. The In operator simply tests whether an object is currently stored in a category or in one of the categories of a category set. The Member operator tests the membership conditions of a category or category set. Member always checks for the satisfaction of the type expression of a category as well as invokes the appropriate classifier if there is a concept definition associated with the category. Normally, Member is used in triggers rather than queries because membership testing is performed before inserting in the database. As examples, the next query selects items which are in at least 2 accepted deals, while the trigger checks the membership of the DealStatus category. The identifier d in the example is a range variable over the Deal category.

```
RETRIEVE All From Item Satisfies
  deals Satisfies-AtLeast 2 (d) (d In Acce-
  ptedDeal)
TRIGGER LooksBad In Deal (d)
  On Insert
  If (d Member RejectedDeal)
  Then display('This looks like a bad deal')
```

Queries can also reference feasible values. Terms involving feasible values can be evaluated as satisfying or possibly satisfying. The latter are evaluated as true if there exists an interpretation satisfying the term. Syntactically, possibly satisfying terms are marked with a question mark as shown in the following example. More details about the semantics of feasible values and their evaluation in queries can be found in[Jung, 1992].

```
RETRIEVE All From Deal Satisfies
  [item.brand = 'Brand1' AND discount >? 0.2]
```

The ExOM provides operators for learning about categories, testing category membership, and explaining category definitions. To learn about a category, a user defines a data set, invokes a learning algorithm using the data set, and possibly tests the accuracy of the resulting concept definitions. In addition, a simulation experiment can be performed by repeating the following sequence: 1) randomly split the data set into a training and test set, 2) learn from the training set, and 3) test the learned concept definitions with the test set. As examples of the learning operators, the first operation specifies a data set for the DealStatus category set using a maximum of 200 randomly selected instances that satisfy the target query. The second operation (LEARN) invokes the learner at the current time (NOW) for DealStatus using a randomly selected 60% of the previously specified data. The third operation (EVALUATE) classifies the remaining part of the data set (here, 40%) using the concept definition (a decision tree) created by the previous LEARN query.

```
DATASET DealStatusSet For DealStatus
 Attributes iname, sname, disc, advertise,
 minord, minadtime
 Instances Existing
 Query
      RETRIEVE item, name, store. name,
      discount, coop-advertise, min-order,
      min-ad-time
          From Deal (d) Satisfies d IN Deal-
          Status
 Method Random
 Maximum Size 200
LEARN DealStatus With DealStatusSet Training
%: 60 Time: Now
EVALUATE DealStatus With DealStatusSet
```

Other operators support membership testing of an object in a category and explanation about why or why not an object is a member of a category. As previously mentioned, testing category membership is implicit when inserting an object. By default, the concept definition is interpreted, and the user is notified if the concept definition is not satisfied. Besides insertion, the user can explicitly test membership in a category and receive an explanation. In the following example, the user asserts an object in the Deal category and asks for an explanation about mem-

bership in the AcceptedDeal category.

```
INSERT deal202  In Deal Satisfies
 [dealno := 202, item := item102, store :=
  retailer2, discount := 0.25,
  coop-advertise := 0.10, min-order := 0,
  min-ad-time := 0]
EXPLAIN deal202 In AcceptedDeal
```

Since the concept definition is a list of weighted feature records, the explanation provides the degree of match about each attribute value as the following text demonstrates; deal202 is a member of AcceptedDeal because its similarity to the following weighted feature record is .99 which is above the match threshold of .66.

```
    [store.name = 'Chain2' (1.0), discount:
     = 0.3 (0.8)]
The details of the similarity comparison are
as follows:
    store.name = 'Chain2' has similarity of 1.0,
    discount = 0.3 has similarity of 0.78
    1.78 is .994 of the maximum similarity of 1.8.
```

In queries about category definitions, a user can request an explanation about the concept definition (entire definition or part of it), classification function, or learning algorithm. The following example demonstrates an explanation of a list of weighted records.

EXPLAIN Definition In AcceptedDeal

AcceptedDeal is an unstructured category with the following list of weighted feature records as a concept definition.

```
Weighted feature record 1:
    item.brand.name = 'topbrand1' has im-
    portance weight 1.0,
    store.name = 'Chain1' has importance
    weight 1.0,
    store.location = 'Denver' has importance
    weight 0.8,
    coop-advertise = 0.2  has importance
    weight 0.75
Weighted feature record 2:
    store.name = 'Chain2' has weight 1.0,
    discount: = 0.3 has importance weight 0.8
```

Ⅲ. Learning and Classification Components

In this section, we present the details of the learning and classification components of the ExOM which were informally described in the previous section. We first begin with formal definitions underlying ExOM databases followed by definitions of classifiers and learners.

3.1 Categories

As in other object-oriented models, ExOM categories provide convenient partitioning of objects for storage and retrieval. In addition, ExOM categories contain concept definitions generated by learning algorithms and used by classification functions. We formally define categories and related aspects beginning with ExOM databases.

Definition 1: An ExOM database is a tuple <O, C, CL, L, M> where

O is a set of domain objects,

C is a set of categories,

CL is a set of classifiers (Section 3.2),

L is a set of learners (Section 3.3), and $M :: 2^C \rightarrow CL \times M$ assigns a classifier and learner pair to a subset of categories where

2^C is the power set (set of all subsets) of category identifiers.

Objects are pairs consisting of an identifier and a feature value. A feature value is an association of labels to values. Formally, ExOM values are drawn from the recursive domain V:

Definition 2:

$V = B_0 \cup B_1 \cup ... \cup B_n \cup O_{ID} \cup FV \cup S$ where

$B_0, B_1, ..., B_n$ are base types

O_{ID} is the set of object identifiers

FV is the set of feature values (finite mappings from the set of labels (A) to values)

$FV = A \mapsto V = \{ fv \in A \rightarrow V \mid \{a \in A \mid fv(a) \in V\} \text{ is finite} \}$

S is the set of set values where a set $S_i = \{v_1, v_2, ..., v_n\}$ with $v_i \in V$

A type is a set of values drawn from V. The ExOM supports type expressions formed from the basic types, range types defined over a base type with an interval specification, enumerated types defined over a base type with a value specification, set types defined with a minimum and maximum cardinality, feature type expressions defined as an association of labels to type expressions, and ExOM category names. The use of a category name implicitly defines a feature type because the denotation of a category includes a feature value. Other type constructors such as function types, lists, and labeled unions can be included but are not further discussed here. Formally, types are defined as expressions that denote a subset of V. Denotation of a type expression T with environment η is defined by $T \| \tau \| \eta$ where $T \in TE \rightarrow 2^V$ is the semantic function for type expressions (TE), τ is a generic type and $\eta \in T_{ID} \rightarrow 2^V$ is an environment, a mapping of type identifiers (T_{ID}) to subsets of V.

Definition 3: a type expression is one of the following:

$T \| k_i \| \eta = B_i$ Where k_i is a type constant (a base type)

$E(v_1, v_2, ..., v_n)$ is an enumerated type with value set $\{v_1, v_2, ..., v_n\}$ where

$T \| E(v_1, v_2, ..., v_n) \| \eta = \{v_1, v_2, ..., v_n \mid \exists B_i (\{v_1, v_2, ..., v_n\} \subseteq B_i)\}$

$RG(\lambda, \upsilon)$ is a range type with minimum(λ) and maximum(υ) values where

$T \| RG(\lambda, \upsilon) \| \eta = \{r \mid \exists B_i (r, \lambda, \upsilon \in B_i \wedge (r \geq \lambda) \wedge (r \leq \upsilon) \wedge (\lambda \leq \upsilon))\}$

$(a_i :: \theta_i)$ is a feature type where

$T \| (a_i :: \theta_i) \| \eta = \bigcap_i \{ fv \in FV \mid fv(a_i) \in T \| \theta_i \| \eta \}$

$ST(\lambda, \upsilon, \theta_i)$ is a set type with minimum (λ) and maximum (υ) cardinalities

$T \| ST(\lambda, \upsilon, \theta_i) \| \eta = S_i$ where $S_i \subseteq T \| \theta_i \| \eta \wedge (|S_i| \geq \lambda) \wedge (|S_i| \leq \upsilon)$

$T \| C_i \| \eta = \{ < o_{id}, fv > \mid o_{id} \in C_i.O \wedge fv \in T \| C_i.ft \| \eta \}$ where

C_i is a category name, $C_i.O \subseteq O$ is the set of object identifiers of C_i, and

$C_i.ft$ is the feature type expression of C_i.

Objects are organized into categories which define conditions for object membership and possibly an interpretation function to classify objects. Formally,

Definition 4: an ExOM category is a tuple $< C_{id}, ft, P, cd_j, cl >$ where

C_{id} is a category identifier

$ft \in FT$ is a feature type x

P is the set of parent categories

$cd_i \in$ CD is an optional concept definition of kind j
$cl \in$ CL is an optional classifier compatible with cd_j.

Most categories are defined with only the first three components. Typically, the concept definition and classifier are specified for category sets rather than individual categories because learning algorithms require training sets with both positive and negative examples, and classifiers choose membership from a collection of categories. As we noted previously, a concept definition is a structure that is generated by a learner and interpreted by a classifier. In order to support a broad range of learners and classifiers, the ExOM is not limited to a single kind of concept definition. We formally define the set of concept definitions CD as below. Note that the definition is capable of incorporating different learning algorithms such as weighted decision tree.

Definition 5: CD = FT + WFVSet + RLSet + DT where
FT = A $\mapsto$ TE is the domain of feature types
WFVSet = 2^{WFV} is the power set of weighted feature values where
WFV = (A $\mapsto$ V × W) × C_{ID} is the domain of weighted feature values where
W is a real number [0..1] indicating the weight or importance of the feature
RLSet = 2^{RL} is the power set of rules where
$rl_i \in$ RL $\wedge$ rl_i = <*lhs*, *rhs*, *st*> where
lhs (left hand side) is a set of triples <label, relop, scal-expr> where
relop is a relational operator and scal-expr is a scalar value, set of scalar values, or pair of scalar values
rhs (right hand side) is a category identifier or pair <label,value>, and
st is a real number [0..1] indicating the strength of the rule
DT is the set of decision trees with $dt_i \in$ DT $\wedge$ $dt_i = \langle node, \{< edge, dt_i >\}\rangle$ where
node is a label for non-leaf nodes and a category identifier or nil for leaf nodes
edge is a scalar value or pair of scalar values (an interval specification).

Practically speaking, care should be exercised to make machine learning algorithms in the ExOM computationally sound. For example, rules are limited to conjunctions of simple comparisons as most machine learning algorithms cannot efficiently search a space of more complex rules. Another complication comes from that some learning algorithms use feature values, set-values, and fuzzy values, enabling a concept definition to indirectly reference attributes of an object. For example, a recursive dot expression of the form, $label_1.label_2...label_n$, represents a hierarchical attribute sequence where $label_i$ refers to an attribute of $label_{i-1}$. This convention of grouping attributes does help improving the learning capabilities of learners but it also can invite computational obstacles.

The ExOM has two kinds of categories, structured and unstructured. The feature type of a structured category mandates objects to satisfy the feature type before being inserted in the category. In addition, the feature type defines the maximal collection of attributes. Each object in a structured category must not have extra attributes except for attributes defined for descendant categories in which the object is a member. Besides the feature type, a concept de-

finition can be given for additional constraints on membership. In contrast with a structured category that adopts a closed world view, unstructured categories support an open world interpretation of feature types to provide more flexibility but less uniformity. If the feature type is given to unstructured categories, it provides a necessary condition for category membership. Formally,

Definition 6: a structured category $<C_{id}, ft, P, cd_i, cl>=$

$\{obj \mid \forall p_i \in P(obj \in p_i \wedge M_{ft}(obj.f,ft) \wedge M_{cl}(obj.f,cd_i) \wedge$ (Attrs(obj)-DirAttrs(DescCats(C_{id},obj))) $\subseteq$ Attrs(ft)

obj.f denotes the feature value of obj,

M_{ft}:: FV × FT → *Boolean* is the membership function for feature types

M_{cl}:: FV × CD → *Boolean* $\cup$ [0..1] is the membership function for classifier *cl*

Attrs :: O $\cup$ FT → 2^A returns the attributes of its argument

DirAttrs :: 2^C → 2^A returns the union of the direct (non-inherited) attributes of each category in its domain

DescCats :: C × O → 2^C returns the descendant categories of C in which O is a member.

Definition 7: an unstructured category $<C_{id}, ft, P, cd_i, cl, i, \tau, \delta>=$

$\{obj \mid \forall p_i \in P(obj \in p_i \wedge M_{ft}(obj.f, ft) \wedge$ Confirm $(M_{cl}(obj.f, cd_i), i, \tau, \delta))$ where

i, τ, δ are an interpretation of the concept definition (necessary, sufficient, or both), the threshold, and the critical range respectively.

The Confirm function in the definition depends on the interpretation value. If the concept definition is necessary, Confirm is true if the membership function returns a value beyond the critical range of the threshold, and false if the value is below the threshold. If the certainty value is close to the threshold, the user is asked to confirm membership. Similarly for sufficient concept definitions, Confirm is true if certainty is above the threshold, false if not close to the threshold, and the user is asked to confirm if the certainty value is almost to the threshold. For necessary and sufficient concept definitions, Confirm returns the same answer as the membership test. Formally,

Definition 7.a: Confirm$(M_{cl}(obj.f, cd_i), i, \tau, \delta)=$

Case i = necessary: $\begin{cases} \text{true if } M_{cl}(obj.f, cd_i) > \tau \\ \text{false if } M_{cl}(obj.f, cd_i) \le \tau \end{cases}$

Case i = sufficient: $\begin{cases} \text{true if } M_{cl}(obj.f, cd_i) > \tau \\ \text{false if } M_{cl}(obj.f, cd_i) < \tau - \delta \\ \text{ASK otherwise} \end{cases}$

Case i = necessary and sufficient: $M_{cl}(obj.f, cd_i)$

In the ExOM, Category sets are the glue between categories and machine learning. A category set defines a collection of categories sharing a learning algorithm, classifier, and concept definition. Formally,

Definition 8: a category set is defined as $<CS_{id}$ C_r, CatTree, cd_j, cl, l, *CatType*> where

CS_{id} is a category set identifier,

C_r is a category identifier that serves as the root of the category set,

CatTree is a tree (nested list) of category identifiers,

cd_j is a concept definition,

$cl \in$ CL is an optional classifier compatible with cd_j,

$l \in L$ is learner compatible with cd_j and *CatType* is either Fixed or Non-Fixed.

A category set can be either fixed or non-fixed. In a fixed category set, each element category cannot have its own attributes nor have any descendant categories except for those defined in the category set. Because of these restrictions, element categories are unstructured and only defined in the category set definition. In a non-fixed category set, the only restriction is that all element categories are either structured or unstructured. The element categories must be separately defined outside of the category set definition. Often, the set of sub concepts does not have clear cut boundaries. For example, the division of customers into price seekers and prestige seekers does not involve any new features for the sub concepts but rather different interpretations of the features of customer. Because of this intended use, element categories in a fixed category set inherit attributes from the root of the category set but may not add or change the inherited set.

3.2 Classifier

In this section, we define the components of a classifier and depict them for the kinds of concept definitions supported by the ExOM beginning with the definition of a classifier.

Definition 9: An ExOM classifier is a tuple $<CL_{id}, M_b, M_g, k,$ PA, EX> where

CL_{id} is a classifier identifier,

M_b is a Boolean membership function of the form

FV × CD → *Boolean* (individual category test) or

FV × CD → C_D (category set test),

M_g is an optional graded membership function of the form

FV × CD → [0..1] (individual category test) or

FV × CD → C_D (category set test) or

k indicates the kind of concept definition

PA is an optional set of parameters used by a membership or explanation function where

$pa_i \in$ PA is a parameter of the form <label, value>

$EX_i \in$ EX is an explanation function (see the discussion below)

Membership testing in the ExOM can be performed in two different contexts, depending on whether a concept definition is defined for a category or category set. In the former case, the concept definition of the category is tested resulting in either a Boolean or graded value. In the latter case, a set of concept definitions is tested resulting in the selected element category with either a graded or Boolean value. In the case of a graded membership function, the category with the highest (thus promising) value is returned. For a Boolean membership function, various strategies are possible and the membership function should use a heuristic to choose a best-fitting category.

An individual classifier includes membership functions for testing either a category or category set. Normally, there are at least two classifiers per concept definition to cover testing of both individual categories and category sets. If a concept definition is not ordered, there are frequently multiple approaches to testing a concept definition. This flexibility enables a designer to select among alternative functions that best match the characteristics of a category of

interest. For example, in classifying census forms about industrial and occupation categories, different membership functions were reported as showing higher accuracy[Creecy *et al.*, 1992].

As a generator of inductive expert systems, the ExOM provides explanation about membership decisions. The basic kind of explanation answers the questions why or why not. While a rule-based expert system often displays the chain of rules used to reach a decision, the ExOM rather returns concept definitions responsible for the classification decision, a detailed explanation of the success or failure, and a certainty value if the classifier uses a graded membership function. Formally,

Definition 10: an explanation function EX_i is of the form

$FV \times 2^{CD_k} \rightarrow CD_k \times String \times [0..1]$ or

$FV \times 2^{CD_k} \rightarrow C_{ID} \times CD_k \times String \times [0..1]$ where the output concept definition (CD_K) is drawn from the input set (2^{CD_k})

For disjunctive concept definitions, the manner in which the responsible element(CD_K) is selected depends on the kind of concept definition and membership function. If the concept definition is ordered such as a decision tree or ordered rules, the first failing or succeeding path/rule is selected. If the associated membership function is graded, the disjunct with the largest value is selected. Otherwise, the most satisfying disjunct can be selected based on from the results of the explanation function. Explanation functions can return information on suitable categories tested for membership rather than only the most satisfying category if more than one category satisfies the concept definition or no categories satisfy it.

3.3 Learner

The ExOM supports two kinds of learning about categories. For the traditional mode of learning, a database designer directly provides concept definitions using editors and browsers for each kind of concept definition. More interestingly and importantly, the ExOM supports a wide range of machine learning approaches, which we elaborate on in this section.

Fundamental to every learning algorithm is the notion of a training set. A training set is a collection of examples (attribute, value pairs) from which a learning algorithm makes inductions[Anthony and Biggs 1992]. The ExOM recognizes supervised training sets (Definition 11) in which every example is associated with a category label and unsupervised training sets without category labels. We require that supervised training sets be functions andthat examples in training sets satisfy constraints of the categories in which they were drawn. Formally,

Definition 11: a supervised training set (sts_i) is defined as $STS = FV \mapsto C_{ID} = \{sts_i \in STS | (fv_j = fv_k) \Rightarrow (sts_i(fv_j) = sts_i(fv_k))\}$

Definition 12: an unsupervised training set (ustsi) is defined as $usts_i \in \cup STS = 2^{FV}$

Learning algorithms use training sets to determine concept definitions, cluster data, and discover interesting relationships. The ExOM recognizes three kinds of learning algorithms

depending on the input (supervised or unsupervised training set) and output (concept definitions and clustering). A supervised concept learner (Definition 11) determines a collection of concept definitions for a supervised training set. A clustered concept learner determines a grouping for an unsupervised training set and possibly concept definitions for the groupings. A discovery learner determines interesting subsets of an unsupervised training set as well as concept definitions for the subsets[3]. A clustered concept learner differs from a discovery learner in that the grouping produced by a clustered concept learner covers the training set while the grouping produced by a discovery learner need not (and typically does not) cover the training set. In addition, the discovery learner also generates concept definitions while the clustered learner need not generate concept definitions. Practically, discovery learners have very large training sets because their purpose is to find interesting patterns in large data sets. Definitions 13 through 15 summarize this discussion.

Definition 13: a supervised concept learner (scl_i) is an element of $SCL = STS \mapsto 2^{CD}$)

Definition 14: a clustered concept learner (ccl_i) is an element of $CCL = UTS \mapsto 2^{UTS} U (2^{UTS} \times 2^{CD})$

Definition 15: a discovery learner (dl_i) is an element of $DL = UTS \mapsto (2^{UTS} \times 2^{CD})$

For a specific learning algorithm, its kind is implicitly defined by a number of properties that must given before it can be used. The properties of a learning algorithm include the kind of concept definition it generates, timing (incremental or batch), category space (flat or hierarchical), supervision level (supervised or unsupervised), and attribute scale (nominal and/or numeric). If the learning approach is unsupervised, a few other properties are relevant including category space size(given and/or learned), disjointness of the category space (disjoint or overlapping), and the covering of the category space (covering or non-covering). Other attributes including attribute cardinality (scalar or set) and value precision (actual or fuzzy) are not currently supported in the ExOM because few machine learning algorithms use set values and fuzzy values. Formally,

Definition 16: an ExOM learner is <L_{id}, *K, T, CS, SL, AT, SZ, DIS, COV*> where
L_{id} is a learner identifier,
K indicates the kind of concept definition,
T indicates the timing (incremental or batch),
CS indicates the category space (flat or hierarchical),
SL indicates the supervision level (supervised or unsupervised),
AT indicates attribute types accepted (nominal, numeric, or both),
SZ is an optional size constraint (given or learned),
DIS is an optional disjointness constraint (disjoint or overlapping), and
COV is an optional covering constraint (cover or non-cover).

Table 1 depicts prominent learning algorithms and their property values. Abbreviations

3) Of course, it is up to the user or expert that moderate the interestingness of the learning process.

<Table 1> Property Values of Selected Learning Algorithms>

Learning Algorithm [Reference]	Learner Properties							
	K	T	CS	SL	AT	SZ	DIS	COV
ID3 [Quinlan 1986]	DT	B	F	S	Nom*	-	-	-
ID5R [Utgoff 1989]	DT	I	F	S	Nom*	-	-	-
IB3 [Aha, *et al.*, 1991]	WFV	I	F	S	All	-	-	-
Unimem [Gennari *et al.*,89]	WFV	I	H	U	All	L	Non	Cov
Cobweb [Gennari *et al.*,89]	WFV	I	H	U	Nom	L	Dis	Cov
CN2 [Clark and Niblett 1989, Clark and Boswell 1991]	Rule	B	F	S	Nom	-	-	-
ITRULE [Smyth and Goodman 1992]	Rule	B	F	U	Nom	L	Non	Non

* Preprocessor transforms numeric into nominal values.

have been used for property values, e.g., B means Batch and I denotes Incremental for the Timing property respectively. Note that supervised algorithms (*SL=S*) do not have values for the size (*SZ*), disjointness (*DIS*), and covering (*COV*) properties.

The ExOM uses these properties to apply a learning algorithm: when to learn, what to learn, and what constrains the learner. The timing property determines whether the learner depends on the order of the training instances[4]. All learners except for ones that only cluster produce concept definitions that are limited to one of the kinds of concept definitions supported by the ExOM. For unsupervised learners, the ExOM uses the size(SZ), category space (CS), covering(COV), and disjointness (DIS) properties to determine a learner's output. Some learners are constrained by the kind of attributes in the training set. If a learner is constrained by numeric or nominal attributes, a learner can only be applied with compatible attributes.

The ExOM provides a number of operators to support the learning process. The learning process begins with the definition of a data set for the learning phase using the DATASET operator. The user specifies the category set, relevant attributes in the data set, the sampling method (random or systematic), the sample size, and the data source (external file, generated da-

4) If the timing is incremental, the order is significant and learning occurs after each example is presented. Training instances are presented in the order in which a learning query produces them. If the timing is batch, the order is not significant and learning does not occur until the entire training set is assembled.

ta set, or data base query using existing or new instances). As a result of the DATASET operator, the specifications about the data set are stored for later use by the LEARN operator, which imports the data set, invokes the specified learner, and saves the result of the learning. If the learner is unsupervised, the data set is split into a training set that is used by the learner and a test set that is used to test the learned concept definitions. The split percentage indicates the percent of the data set used in the training set. The split percentage is 100% for unsupervised learners. The EVALUATE operator interprets the concept definitions created by a supervised learning algorithm against a test set and reports learner's accuracy. Last but not the least, the EXPERIMENT operator repeatedly executes the LEARN/EVALUATE sequence for a specified number of trials and accumulates the EVALUATE results. It is used only for supervised learners.

Ⅳ. Consistency of ExOM Databases

In strongly-typed object-oriented databases, taxonomic reasoning provides automatic classification of a new concept in a hierarchy of concepts and ensures the consistency and minimality of the conceptual schema. A well behaved learning process uses background knowledge and a training set to generate concept definitions for a collection of categories. Background knowledge typically consists of a collection of feature type expressions, one for each category. The background knowledge for a category will also include any concept definitions from parent categories. In this case, the background knowledge is the conjunction of the feature type expression and inherited concept definitions. Recall that a supervised training set is a collection of objects where each object consists of a set of values for the predictor attributes and a category identifier representing one of the categories in which to learn. The training set is produced by a data generator where each object generated adheres to its associated background knowledge. Furthermore, the training set is used by a supervised concept learning algorithm to generate minimally consistent concept definitions. Formally, a data generator DG and minimally consistent concept learner are defined as below.

Definition 17: DG = $2^{BG} \mapsto$ STS where $bg_i \in$ BG = $te_i \wedge cd_i$ where $te_i \in$ TE and $cd_i \in$ CD with background knowledge te_i and cd_i for category i such that
$\forall dg \in$ DG, $\forall bg \in 2^{BG}$, $\forall sts_i$
$\in dg(bg)$ (satisfies(sts_i, bg_k)) where
k = CatIndex(sts_i) and
CatIndex(x) returns the index of the category of training example x.

Definition 18: A supervised concept learner $scl_i \in$ STS $\mapsto 2^{CD}$ generates minimally consistent concept definitions if and only if $\forall sts \in$ STS, $\forall cd_j \in scl_i(sts)(\exists sts_k \in sts)$(satisfies($cd_i$, sts_k))).

Definitions 17 and 18 restrict the data generation process and the output of supervised concept learners. Definition 17is a simplification of a data generation process. An actual data generation process usually involves either an

historical or hypothetical data set. In the former case, the objects of the training set are taken directly from an existing category in a database. In the latter case, hypothetical objects are defined usually in consultation with a group of experts. We assume that adequate care is exercised so that each object in the training set satisfies its associated background knowledge in the form of a feature expression possibly augmented with inherited concept definitions from parent categories. Definition 18 establishes that concept definitions generated by a learner need not fully explain the training set. However, we require that an arbitrary concept definition cdj be consistent with at least one element of the training set sts_k.

Definitions 17 and 18 lead to several simple propositions regarding the relationship between a concept definition generated by a learner and the backgroundknowledge. First, in a well-behaved learning process, a supervised concept learner generates concept definitions that are consistent with the background knowledge (i.e., feature type expression and inherited concept definitions) of the category. By Definition 18, each concept definition is satisfied by at least one member of the training set and by Definition 17, each example in a training set satisfies its background knowledge. Thus, the background knowledge is consistent with the generated concept definition because there is at least one training example that satisfies both. Second, in a well-behaved learning process, a concept learner generates concept definitions that do not necessarily imply the background knowledge of their category. A feature type expression includes all of the attributes of a structured category and a subset of the attributes of an unstructured category. For both structured and unstructured categories, a concept definition may only include a proper subset of the attributes from a feature type expression. Thus a concept definition containing a proper subset of the attributes cannot possibly imply a feature type expression with conditions on all the attributes.

However, in practice, supervised concept learners generate concept definitions that may not imply their associated feature type expression. By design, supervised concept learners attempt to compress their training set because they try to remove attributes that do not significantly contribute towards good classification performance and they prefer simple concept definitions (few terms) to more complex ones, other things being equal. Unfortunately to our proposal, there could be case where a feature type, usually defined by the database designer, may be in a collision course with one or more concept definitions generated by a learning algorithm. To prevent such conflicts, we require that an inherited attribute cannot be redefined if it appears in the concept definition of an ancestor class. Generally this requirement should not be a problem because the concept definition typically imposes tight restrictions on any relevant attribute and further restrictions in descendant categories are usually not meaningful. We formalize the consistency issue as below.

Definition 19: Attribute Redefinition Rule:
$\forall\, a_i \in ft_c (a_i \in cd_p \rightarrow ft_p.a_i = ft_C.a_i)$ where
C is a child of category P,
cd_X is the concept definition of category X, and
$ft_X.a_i$ is the feature type expression of cat-

egory X restricted to attribute a_i

Proposition 1 follows from Definitions 17, 18, and 19. Proposition 1 states that a parent category subsumes a child category if both concept definitions are generated by restricted concept learners, the feature type of the child category satisfies the Attribute Redefinition Rule with respect to the concept definition of the parent category, and SUBS determines that the feature type expression of the parent category subsumes the feature type expression of the child category. Formally,

Proposition 1: P SUBSUMES C where
P = $cd_p \wedge ft_p$ and C=$cd_p \wedge cd_c \wedge ft_c$ iff
1. $cd_p = scl_i(dg_j(ft_p))$ where $scl_i \in$ SCL and $dg_i \in$ DG,
2. $cd_c = scl_m(dg_n(ft_c \wedge cd_p))$ where $scl_m \in$ SCL and $dg_n \in$ DG,
3. ft_c satisfies the Attribute Redefinition Rule with respect to cd_p, and
4. SUBS(ft_c, ft_p).

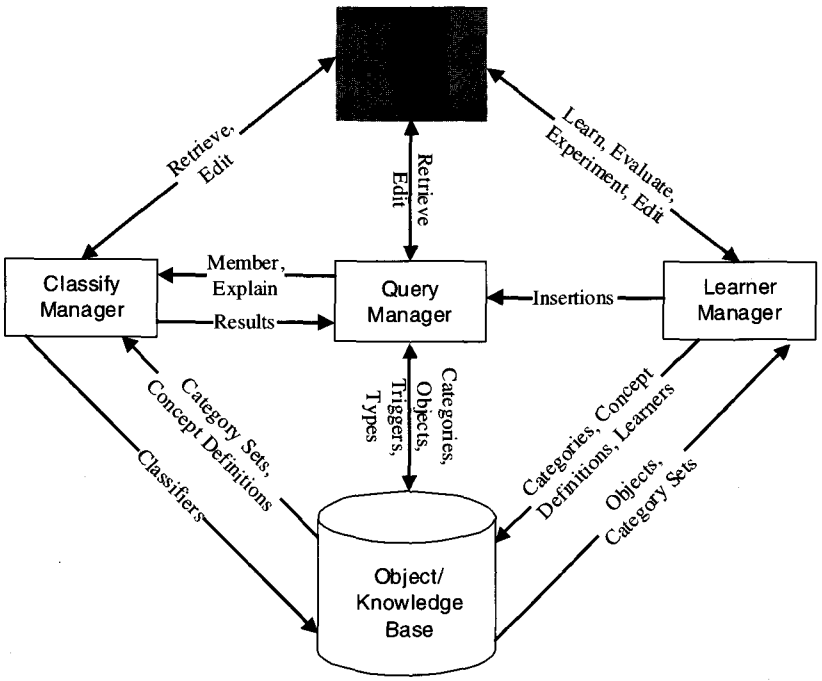
The proof constructively demonstrates that the definition of C logically implies the definition of P. First, by the fourth condition, $ft_c \Rightarrow ft_p$. Second, $ft_c \wedge cd_p \Rightarrow ft_p \wedge cd_p$ thanks to the Attribute Redefinition Rule and Definition 17. It follows from Definition 18 that $ft_c \wedge cd_p \wedge cd_c \Rightarrow ft_p \wedge cd_p$ as cd_c is consistent with $ft_c \wedge cd_p$ because cd_c is satisfied by at least one training example that also satisfies $ft_c \wedge cd_p$.

Based on Proposition 1, we propose a strategy to test a new category for consistency with its parent categories. When adding a category, the attribute redefinition rule and subsumption testing is performed with respect to every parent category. If a learner is invoked for a new category, a background check is enforced for every data set used where the background knowledge is the feature type expression and the inherited concept definition. Every subsumption relationship given by a designer can be verified using this approach, but some subsumption relationships cannot be deduced using Proposition 1 because the Attribute Redefinition Rule is a sufficient but not necessary condition for subsumption.

With taxonomic reasoning, many categories must be represented as primitives because the class description expression will not usually be a necessary and sufficient condition of class membership. Some subsumption reasoning approaches (e.g. [Borgida *et al.*, 1989]) have a safety valve to deal with this situation. We note that most rule languages in active database systems are not amenable to taxonomic reasoning. In addition, even the conjunction of a feature type expression (a necessary condition) and a concept definition are usually not necessary and sufficient conditions of category membership because the true concept definitions remain unknown.

V. Prototype Implementation

The architecture of the ExOM contains three major components as depicted in Figure 2. The Query Manager is the interface to the objects, categories, and types of an ExOM database. Users can retrieve objects by their content, define categories and types, and indirectly invoke Classifier Manager when asking about category membership. The Classifier Manager supports the definition and retrieval of classifiers as well

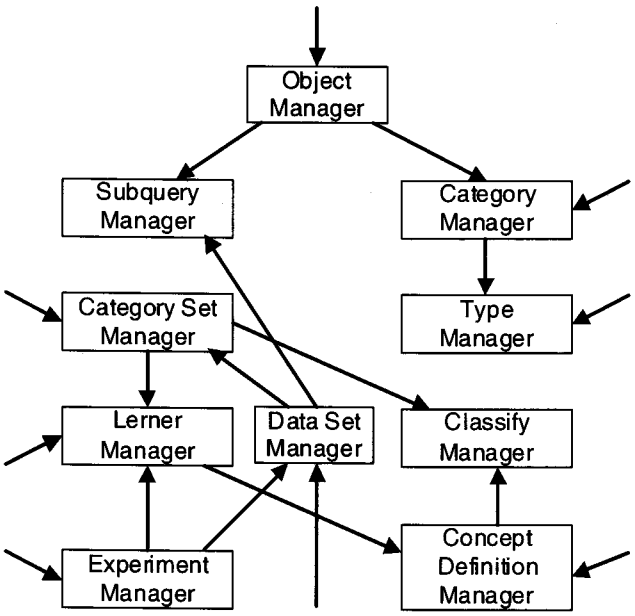


<Figure 2> ExOM Architecture

as testing membership of an object in a given category and explaining its decision. The Learner Manager acquires objects, invokes learning algorithms, and creates concept definitions. In the case of unsupervised learning, it also creates categories and inserts objects into those categories. The knowledge base contains definitions of categories, types, concept definitions, triggers, learners, and classifiers.

A prototype with most of the features described in Sections 2 and 3 has been implemented in Smalltalk environment. The object-orientation and weak-typing of Smalltalk allows us to develop the prototype in a faster way. In the prototype, the Query Manager (Figure 3) is divided into an object manager for manipulating and querying objects, a type manager for defining named type expressions, and a category manager for defining categories and teacher supplied concept definitions. The Classifier Manager has been divided into managers for concept definitions and associated membership and explanation functions (classifiers). The Learner Manager has been divided into a learner manager for registering learners, a category set manager for grouping categories with a learner and classifier, a data set manager for creating data sets from queries, external files, or a data generation program, and an experiment manager for executing the learn, test, and experiment operators.

All three concept definitions (weighted feature records, weighted rules, and decision trees) have been implemented. For the first two, there is an associated classifier for single categories (teacher supplied) as well as for category sets (learner generated). Several learning algorithms



<Figure 3> Flow among ExOM Manager

have been implemented: IB3[Aha *et al.*, 1991], CN2[Clark and Niblett 1989, Clark and Boswell 1991], and several versions of ID3[Quinlan 1986]. A number of features have been omitted due to time restrictions including feasible values, the possibility operator, limited query capabilities (only conjunctive semi-joins), support for unstructured categories, and unsupervised learning algorithms. We are in the process of converting the object and category manager to a suitable object-oriented database engine to improve performance.

The ExOM prototype is invoked by interactively navigating among a collection of window interfaces (one for each manager). As a result, the syntax in Appendix A has been implemented through interactive specification rather than as a translator of language strings. Figure 2 depicts the flow among interface managers. For flexibility in defining a knowledge base, links between managers are supported in the form of popup menus. For example when defining a category set, the user can easily branch to the learner and classifier managers. The arrows without a source mean that the manager can be invoked directly from the ExOM main menu. Note that the sub query manager (not mentioned above) handles category navigation in queries. To demonstrate the flavor of the prototype, Figure 3 illustrates the object manager when the Deal category has been selected and its attributes displayed. The window is divided into fixed size panes to select a category, select a field (right most list pane), select a comparison operator (adjacent list pane), enter a value (adjacent list pane), and view results and messages (bottom text pane). Within each pane, there is a popup menu (not shown) to perform

various functions. For example in the category pane, the popup menu supports inserting, deleting, retrieving, and updating objects as well as branching to other managers. In the attribute pane, the popup menu depends on the choice selected in the popup menu of the category pane. When inserting an object, the membership of an object can be tested against its concept definition and an explanation can be displayed.

The prototype implementation comprises about 45 new classes, extensions to 10 existing classes, 200 methods, and 15,000 lines of code including comments. As with many window-based programs, much of the code is interface management. Since the syntax is managed interactively rather than froma translator, some of this interface code performs semantic actions rather than just user control and parsing. To demonstrate the ExOM, several small databases have been created including the marketing promotion database described in this paper.

Ⅵ. Related Work

The ExOM has been inspired by work in organization theories and knowledge discovery. Work about information processing in organizations ([Daft and Weick 1984] and [Daft and Lengel 1986]) and organizations as loosely coupled systems [Orton and Weick 1990] has been a general motivation of our research. Work in knowledge discovery has been a more specific influence on our work. As [Frawley *et al.*, 1991] defines knowledge discovery as "the nontrivial extraction of implicit, previously unknown, and potentially useful information from data," the definition extends notions of discovery in machine learning, databases, statistics, and visualization. The ExOM can be viewed as a tool for discovering knowledge in an object-oriented database using a broad range of learning and classification approaches.

From a data model perspective, the ExOM is an extension of object-based models. Object-based models such as [Lalonde *et al.*, 1986], [Lieberman 1986], and [Sciore 1989] provide a more flexible notion of inheritance than class-based data models such as [Mannino *et al.*, 1990] and [Kim 1990]. The ExOM combines both notions (class-based and object-based) through structured and unstructured categories. The focus in this paper is the integration of learning and classification with categories rather than flexible inheritance. Inheritance and other aspects of the ExOM are reported in[Jung 1992].

The ExOM extends the reasoning capabilities of deductive databases [Hansen and Widom 1992] and utilizes learning for more than monitoring database designs. The use of learning and classification in the ExOM is not limited to database design as described in [Ioannidis *et al.*, 1992], [Borgida and Williamson 1985], and [Li and McCleod 1989]. Rather, the ExOM is a general purpose tool in which the user can determine the application of the learning and classification components including to monitor a database design.

Ⅶ. Conclusion

We presented an overview of the Extensional Object Model (ExOM) and described in detail its support for classification and learning. The fundamental motivation of the ExOM is to in-

corporate a broad range of classification and learning methods rather than a single canonical method. By integrating the learning and classification components through grouping of categories for learning and classification, users of the ExOM worry less about the management of discovered knowledge and focus more on a given task. To ensure the consistency of ExOM databases, properties relating concept definitions and feature type expressions were presented. We also briefly depicted a prototype implementation in Smalltalk that supports most of the features described in the paper.

There a few research avenue ahead of the ExOM. The first would be how to improve its capability as a decision making tool. An important short-term extension is an expanded experiment operator. The current experiment operator supports iteration of training and testing with classification accuracy as the performance measure. To satisfy more detailed decision making situations, the experiment operator should support properties of attributes and categories (e.g., costs and noise levels), various performance measures (e.g., information theoretic measures and costs), data set factors (e.g., noise generation and skewness), and experimental factors (e.g., learning algorithms and cost variance levels). Further extensions of the learn, evaluate, and experiment operators would be necessary for unsupervised learning.

Sensitivity analysis capabilities are long-term extensions to the ExOM that are necessary to make machine learning a widely-used analysis tool. In order to better understand a concept definition, a decision maker may want to know the effect on the output when an attribute changes its value such that one or more terms in a concept definition change state (True to False or vice-versa). If the concept definition is deterministic, an explanation function can be implemented as a form of Boolean sensitivity analysis based on the notion of a Boolean derivative. If the concept definition uses weights, the Boolean derivative does not directly apply although the derivative of fuzzy valued functions might. A second kind of sensitivity analysis is output oriented. A decision maker may want to know how to change the values of the attributes to achieve a desired change of outcome. For example, if the output is the prediction of a category representing a firm's bond rating, the decision maker may want to know how to improve the bond rating. Further, if there are costs associated with changing the states of attributes, the decision maker may want to know the least cost way to change categories.

With these extensions and further usage, the ExOM can become an important tool to explore the integration of machine learning and object- oriented databases. We believe that this integration can increase the value of object-oriented databases and lead to improved decision making through wider usage of machine learning.

〈References〉

[1] Anthony, M. and Biggs, N. Computational Learning Theory, Cambridge University Press, 1992.

[2] Aha, D., Kibler, D. and Albert, M. "Instance-Based Learning Algorithms," *Machine Learning*, Vol. 6, 1991, pp. 37-66.

[3] Borgida, A., Brachman, R., McGuinness, D., and Resnick, L. "CLASSIC: A Structural Data Model for Objects," *in Proc. ACM SIGMOD Conference*, May 1989, Portland.

[4] Borgida, A. and Williamson, K. "Accommodating Exceptions in Databases and Refining the Schema by Learning from Them," *in Proc. of the 11th International VLDB Conference*, August 1985, Stockholm, pp. 72-81.

[5] Clark, P. and Boswell, R. "Rule Induction with CN2: Some Recent Improvements," *in Proc. Machine Learning - European Working Session on Learning*, Porto, Portugal, Springer-Verlag, March 1991, pp. 151-163.

[6] Clark, P. and Niblett, T. "The CN2 Algorithm," *Machine Learning*, Vol. 6, No. 4, 1989, pp. 261-283.

[7] Creecy, R., Masand, B., Smith, S., and Waltz, D. "Trading MIPS and Memory for Knowledge Engineering," *Communications of the ACM*, Vol. 35, No. 8, August 1992, pp. 48-64.

[8] Ceri, S. and Widom, J. "Deriving Production Rules for Constraint Maintenance," *in Proc. of the Sixteenth International Conf. on Very Large Data Bases*, Brisbane, Australia, August 1990, pp. 566-577.

[9] Daft, R. and Weick, C. "Towards a Model of Organizations as Interpretation Systems," *Academy of Management Review*, Vol. 9, No. 2, 1984.

[10] DL86 Daft, R. and Lengel, R. "Organizational Information Requirements, Media Richness and Structural Design," *Management Science*, Vol. 32, No. 5, 1986.

[11] Fayyad, U., Piatetsky-Shapiro, G. and Smyth P., "From Data Mining to Knowledge Discovery in Databases," *AI Magazine*, Fall 1996, pp. 37-54

[12] Frawley, W., Piatetsky-Shapiro, G., and Matheus, C. "Knowledge Discovery in Databases: An Overview," *in Proc. First International Conference on Knowledge Discovery and Databases*, October 1991, New York.

[13] Goebel M., Le Gruenwald. "A Survey of Data Mining and Knowledge Discovery Software Tools," *ACM SIGKDD Explorations Newsletter*, Vol. 1, 1, 1999, pp. 1-20

[14] Gennari, J., Langley, P., and Fisher, D. "Models of Incremental Concept Formation," *Artificial Intelligence*, Vol. 40, 1989, pp. 11-61.

[15] Hansen, E. and Widom, J. "Rule Processing in Active Database Systems," *in Advances in Database and Artificial Intelligence*, JAI Press, Greenwich, Connecticut, 1992.

[16] Ioannidis, Y., Saulys, T., D. and Witsitt, A. "Conceptual Learning in Database Design," *ACM Transactions on Information Systems*, Vol. 10, No. 3, July 1992, pp. 265-294.

[17] Jung, C. A Framework for Computer-Supported Interpretation Systems, Ph.D. Dissertation, The University of Texas at Austin, Department of Management Science and Information Systems, May 1992.

[18] Kim, W. "Object-Oriented Databases: Definition

and Research Directions," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 2, No. 3, September 1990.

[19] Lalonde, W., Thomas, D., and Pugh, D. "An Exemplar Based Smalltalk," *in Proc. OOPSLA Conference*, October 1986.

[20] Li, Q. and McCleod, D. "Object Flavor Evolution Through Learning in an Object-Oriented Database System," *in Proc. of the 2nd International Conference on Expert Database Systems*, L. Kerschberg (ed.), 1989, Benjamin Cummings, Menlo Park, CA, pp. 469-495.

[21] Lieberman, H. "Using Prototypical Objects to Implement Shared Behavior in Object Oriented Systems," *in Proc. OOPSLA Conference*, October 1986.

[22] Matheus, C.J., Chan, P.K., and Piatetsky-Shapiro, G., "Systems for Knowledge Discovery in Databases," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 5, No. 6, December 1993, pp. 903-913

[23] Mannino, M., Choi, I., and Batory, D."The Object-Oriented Functional Data Lanaguage," *IEEE Transactions on Software Engineering*, Vol. 16, No. 11, November 1990, pp. 1258-1272.

[24] McCann, J. and Gallagher, J. Expert Systems for Scanner Data Environments, International Series in Quantitative Marketing, Kluwer Academic Publishers, 1990.

[25] Orton, J. and Weick, K. "Loosely Coupled Systems: A Reconceptualization," *Academy of Management Review*, Vol. 15, No. 2, 1990.

[26] Quinlan, J. "Induction of Decision Trees," *Machine Learning*, Vol. 1, 1986, pp. 81-106.

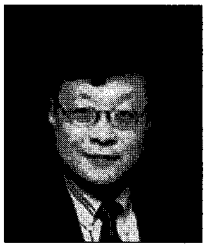
[27] Rao, Raghav and An, Joon M., "The effect of team composition on decision scheme, information search, and perceived complexity," *Journal of Organizational Computing and Electronic Commerce*, 1995, Vol. 5 Issue 1, pp. 1-20.

[28] Sciore, E. "Object Specialization," *ACM Transactions on Information Systems*, Vol. 7, No. 2, 1989.

[29] Smyth, P. and Goodman, R. "An Information Theoretic Approach to Rule Induction from Databases," *IEEE Transactions on Knowledge and Data Engineering*, Vol. 4, No. 4 (August 1992), pp. 301-316.

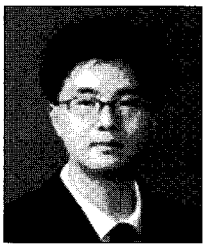
[30] Utgoff, P. "Incremental Induction of Decision Trees," *Machine Learning*, Vol. 4, 1989, Kluwer Academic Publishers, pp. 161-186.

◆ 저자소개 ◆



김용재 (Kim, Yong Jae)
서울대학교 경제학과를 졸업하고, 미국 State University of New York at Stony Brook에서 경제학 석사, University of Kansas에서 전산과학 석사, University of Washington에서 경영학 박사를 취득하였고, University of Colorado Denver에서 조교수를 역임하였으며, 현재는 건국대학교 경영 경영정보학부 부교수로 재직하고 있다. 주요 관심분야는 데이터 서비스 가격결정, 객체지향 모델 구축 및 지식 경영 등이며, 경영정보학 연구, 데이터베이스저널, Operations Research Letters 등의 학술지에 논문을 게재하였다.

안준모 (An, Joon M.)
연세대학교 경영학과 학사, 미국 Texas A& M University, College Station 경영전산 석사, 뉴욕주립대(버팔로 캠퍼스)에서 경영정보 전공으로 박사학위를 수여하였으며 건국대학교 경영대학 경영정보 전공 교수로 재직 중이다. LG-EDS컨설팅 책임컨설턴트, 모토로라 유니버시티 프로젝트 관리 분야 전문 교수, 건국대학교 CIO, University of California, San Diego 연구교수를 역임하였다. 국내 IT아웃소싱 분야 자문 및 사외이사로 활동하고 있다. 연구관심 분야는 IT서비스와 아웃소싱, 글로벌 소프트웨어 산업 등이며 Journal of Organizational Computing, 경영정보학연구 등에 연구논문을 발표하였다.



이석준 (Lee, Seogjun)
고려대학교 산업공학과에서 학사와 석사 학위를 취득하였고 University of Wisconsin에서 산업공학 박사학위를 취득했다. 현재 건국대학교 경영정보학과 부교수로 재직하고 있다. 주요 관심분야는 정보화 성과관리 및 평가, Enterprise Architecture, 정보기술 관리, eHealth 등이다.

◈ 이 논문은 2006년 10월 24일 접수하여 1차 수정을 거쳐 2007년 2월 12일 게재확정되었습니다.