Multiple Overlapping Classifications: Issues and Solutions

Cédric Raguenaud, Jessie Kennedy
School of Computing
Napier University
10 Colinton Road, Edinburgh, EH10 5DT
{c.raguenaud, j.kennedy}@napier.ac.uk

Abstract

This paper discusses issues and solutions for supporting multiple overlapping classifications in database systems. These classifications are commonly found in science, although they are often ignored in computing applications for scientific data, and inappropriate solutions adopted as their replacement. Known database models and classification techniques offer some degree of support for multiple overlapping classifications, but do not fully support the basic features we have identified as necessary: trees/graphs, traceability, semantics of classifications, independence of classification and data, and identity of classifications.

The approach to the problem adopted by the Prometheus project, based on an extended object-oriented database model and the independence of classification schemes from classified data, is presented and discussed.

1. Introduction

Classification is a widespread concept that helps categorise, and therefore simplify data or objects in order to facilitate their understanding and manipulation. Through representing the relationships between classified things, classifications may provide new insights into the things being classified, e.g. discovering that two groups thought to be independent are in fact related in some way or deducing from relationships between groups that they have similar properties. They also allow automatic reasoning, e.g. propagation of attributes in computing models [32] [15] or support user interactions, e.g. simplification of searches.

Examples of familiar classifications include library catalogues where books are placed into categories (e.g. genre) in order to ease access and simplify their management. Medical classification mechanisms, such as the International Classification of Diseases (ICD), which catalogues and relates diseases in order to make prevention, diagnosis, and cure possible. Classification of living organisms as found in for example plant and animal taxonomy and virology.

In order to model and support classifications, we need to understand how they work. Two aspects of classifications can be distinguished: the way objects are grouped into classes of equivalence, and the way classes of equivalence relate to each other and form hierarchies.

Classes of equivalence

In classifications, collections of objects are gathered and classes of equivalence are created within the collection (Figure 1 a-d) via an equivalence function. The equivalence function results in objects being *instances of* their class(es) and not of others.

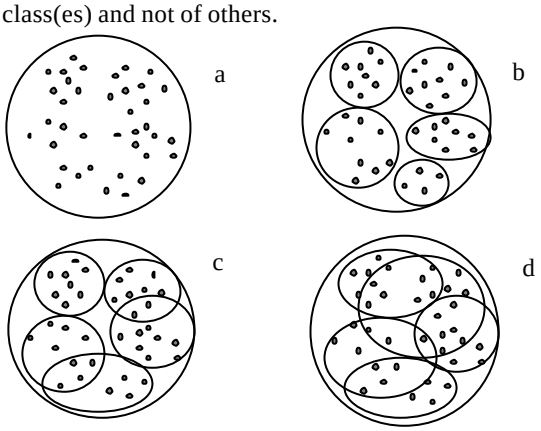

Figure 1: Creation of classes of equivalence

The equivalence function that partitions an application domain dictates how the set of objects to be classified will be clustered. The application of that function may result in the creation of distinct groupings without overlap. This is the case when the equivalence function offers exclusive partitioning. For example, eye colour offers an exclusively partitioning function: people normally have only one eye colour. Applying this equivalence function to a group of objects will result in several distinct sets as shown in Figure 1-b where each set represents a class of

equivalence containing people with a particular eye colour.

In other cases, the partitioning function doesn't provide a clean clustering and objects may belong to several classes of equivalence. For example, the attribute genre of a library classification may have values "fiction", "crime", "drama", and "historical ". This equivalence function may lead to non-exclusive partitioning of the domain because its values are not exclusive. For example, a novel may be of genre "crime" and "historical" at the same time as shown in Figure 1-c. Therefore a book may belong to more than one category simultaneously. It may be argued that this shows a bad choice of equivalence function, but the fact is that these classifications do exist, are seen to be useful and therefore cannot be ignored.

The complexity of the equivalence function may also lead to non-exclusive partitioning of the domain. When the equivalence function is composite, i.e. it is made of several equivalence functions applied simultaneously (the classification is said to be polythetic, it is monothetic when only one function is used), objects to which the function is applied may fulfil the requirements of several classes simultaneously. For example, a book partitioning that takes into account the genre, the period, and the place of origin of the author of books will lead to a partitioning where a book may appear in, say, XXth century writing, crime, and Scottish writing (Figure 1-d).

Class hierarchies

In addition to gathering objects, classes of equivalence can be related to each other via the classification scheme. The mechanism underlying all classification schemes is membership. Indeed, all classification hierarchies imply that lower level classes are members of higher classes. They also imply that objects that are instances of lower classes of equivalence are also, through transivity of the classification function, instances of higher classes of equivalence (Figure 2).

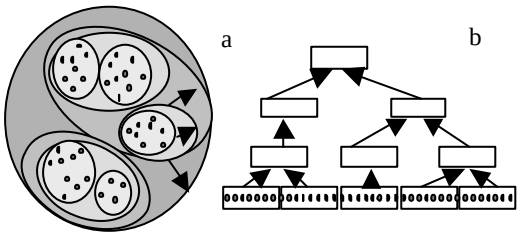

Figure 2: Two views of a classification hierarchy

We can identify at least three main kinds of classification semantics (the classification function): subsumption classifications (is-a), decomposition classifications (part-of), and similarity classifications.

Is-a classifications are based on the concept of specialisation/generalisation. Classes of equivalence are organised into a directed acyclic graph (DAG) where their semantics are specialised in a top down fashion. An example of an is-a classification is the concept of object class hierarchy in computing, where objects are grouped into classes according to their structure and classes arranged into a specialisation/generalisation hierarchy based on e.g. attribute and structure. Another example is library information management, where the classification scheme may be based on genre. Books are placed into categories as shown in Figure 3. The generalisation/specialisation hierarchy places most specific classes at the bottom and more general ones at the top.

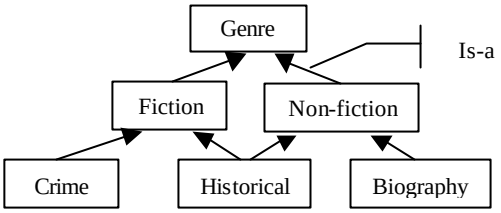

Figure 3: Is-a classification

Part-of classifications are based on the concept of decomposition of objects into smaller objects. Each object is related to its parent (whole) via a part-of relationship. These classifications form an abstract to concrete classification where higher concepts are transitively made of lower concepts. The semantics of part-of relationships have been extensively studied and include functional relationships, topological relationships, and homeomeric relationships [29]. An example of part-of classification is the International Classification of Diseases (ICD), where one classification is based on topology, i.e. it decomposes the human body into sub-parts in order to describe the illnesses that may affect each of them (Figure 4).

Similarity classifications, as other classifications, are based on the concept of membership. The classification function clusters classes of equivalence by similarity. They are related to is-a classifications, that also group classes by similarity (attribute structure) but do not imply the subsumption rules is-a classifications exhibit. An example of similarity classification is plant taxonomy where taxonomists create classifications based on similarity of specimens or *taxa*[1] (the classes of equivalence) according to phenotypic descriptions. Figure 5 shows an extract from a taxonomic classification built on similarity. The classification shows that "Caucalideae" and "Coriandreae" are similar in some respect and therefore belong to the same higher group, "Multiiudatae".

[1] Group of specimens or other *taxa*

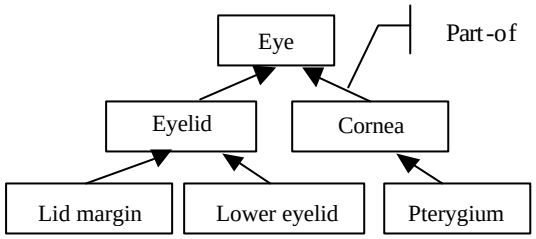

Figure 4: part-of classification

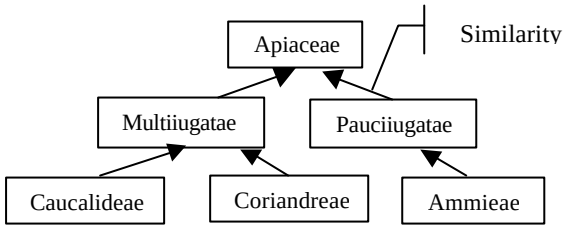

Figure 5: similarity classification

Other classification semantics are possible, as long as the classification function is transitive and a partitioning function for the domain is available.

Plant taxonomy

Plant classification is called a taxonomy because classes appear only once and in one place in each individual classification. Figure 3 shows a non-taxonomic classification because the class "historical" appears both in classes "Fiction" and "Non-fiction".

Plant taxonomy classifications exhibit additional peculiar properties. Plant taxonomy can be called a population-based classification mechanism. Theoretically [18], taxonomic classifications consist of one-level classifications where specimens are put into *piles* (classes) according to their characteristics. The classes (*taxa*) are in addition objects in their own right that are published when a name is assigned to them by application of nomenclature rules [18].

However, for practical reasons, i.e. it is hard to handle thousands or more specimens at once, these one-level classifications are merged into n-level classifications where elements of each level are *instances of* elements of the next higher level, i.e. *taxa* are made members of higher *taxa.* Specimens are members of all higher *taxa* by transitivity of the inclusion relationship.

This leads to classes that are both classes of equivalence that gather objects that fulfil certain requirements (e.g. they look similar), and act as surrogate objects for the objects they recursively contain, therefore are in turn classified.

Multiple classifications

In some application domains, revision of classifications is regular or common. For example, the ICD is revised by the World Health Organization every 10 years approximately and updated every year for minor changes. Each revision creates a new classification of diseases that replaces older ones, and each new version shows the history of classes (e.g. two new classes may correspond to a single class in an older version of the ICD).

In plant taxonomy, revisions are common as they are made when new data, new techniques, or new opinions appear and may lead to a different understanding of the world (e.g. DNA sequencing in the last few years). Unlike for the ICD, new revisions do not replace older ones therefore all classifications ever published are valid. These classifications also form the basis of new classifications through revision. This leads to a large number of classifications (hierarchies of *taxa*) of the same specimens or *taxa* as overlapping groups of specimens (i.e. groups sharing specimens). Figure 6 shows an example of multiple classification in plant taxonomy. The first classification (top), Berchtold & Presl 1820, is highlighted and all the classes that appear in that classification are highlighted in two subsequent classifications, Koch 1824 and De Candolle 1830. It is apparent that groups are moved around and are classified differently over time.

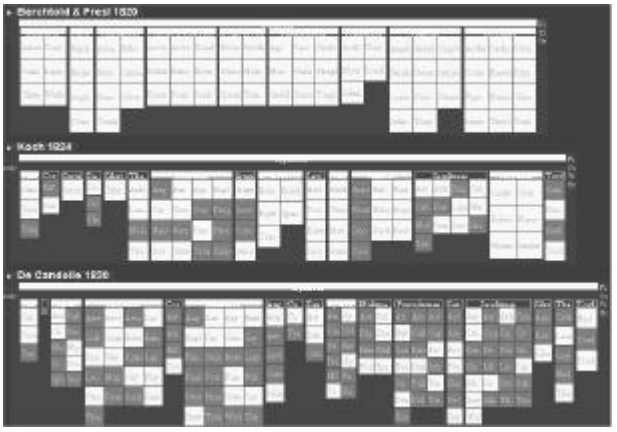

Figure 6: Multiple classifications

As we have seen, classifications are not always the straightforward unambiguous tree structures shown in Figure 2. We have seen that the objects that are classified may appear in many classes of equivalence when the partition function allows it. We have also seen that classes of equivalence can be related in several ways (one of which is is-a) and may appear in several higher classes (non-taxonomic classifications). We have also seen that some application domains generate several classifications overlapping in terms of classified objects and classes (e.g. plant taxonomy, ICD).

Because of the complexity of dealing with these multiple overlapping classifications, they are generally ignored in both biology (e.g. in plant taxonomy where consensus classifications are forced upon taxonomists

[36]) and computing (e.g. where classifications are declared suitable for biological classifications [28] when they only handle single classifications).

This paper first discusses the issues associated with supporting multiple overlapping classifications such as those found in plant taxonomy [36]. Then the ability of the major existing families of database models to handle multiple overlapping classifications is discussed. Our approach (Prometheus) to dealing with the issues is presented in section 4, and we conclude in section 5. A full survey and specification of Prometheus can be found in [37].

2. Issues

There are several issues associated with the handling of multiple overlapping classifications, which were identified during work on the Prometheus project [35]. Although raised in the context of plant taxonomy they apply to classification schemes in general. The first issue is the handling of single classifications, which includes the semantic of the classification mechanism, traceability of decisions and independence of things from their classification. The second is dealing with multiple overlapping classifications including identity of individual classifications and their interconnection.

In order to support multiple overlapping classifications, single classifications must be handled in a fashion that allows the recording of all the information necessary to describe the classification. Classifications in which the things can belong to only one category are tree structured, whereas those where the things appear in several categories are graph based. In addition the semantics of the classification relationships vary amongst classifications: is-a classifications, (e.g. classification of object-oriented programming language classes or library catalogues); similarity classifications (e.g. plant taxonomy); part-of (e.g. component-part classifications); other types of classifications, where the relationships between classes can be anything, e.g. a path that described costs of medical procedures [7], or ontologies. Therefore the selection of the appropriate classification representation for a given domain is important and the implementation of the chosen representation is not trivial, as will be seen in section 3. As a consequence of classification, traceability becomes fundamental. Traceability allows the explanation, in the data, of the motivation for a particular classification. For example, a plant taxonomist should be able to explain why a particular *taxon* has been placed in another. The ICD assigns unique numbers to diseases and operations based on their path in the classification (each branch of the classification carries a number). If classes appear in several placed in classifications, it is not sensible to make that unique number part of the class definition (one class may have several numbers depending on the path used to reach it).

The handling of multiple overlapping classifications requires mechanisms that are not necessary when only single classifications are supported. Firstly we need to be able to identify each classification within the system. If the multiple overlapping classifications result from the repeated classification of the same things then overlaps occur in the classifications and it is necessary to be able to identify those overlaps.

An important feature of classification is that things should be independent from their classification. In the real world anything can be classified, not only things deemed classifiable. It would make no sense to design things so that they can be classified and need to maintain information about their classifications. Moreover, mixing the description of things with their ability to be classified would increase their complexity and reduce reuse and maintainability. This situation is exacerbated when things are multiply classified. It is therefore important that the objects that are classified do not participate directly in the classification process. This makes the management of basic data and the activity of classifying independent processes.

In summary, the requirements of a computer system to support multiple overlapping classifications are:

- support for trees/graphs
- support for semantic relationships
- support for traceability
- identification of distinct overlapping classifications
- orthogonality of classification and data

3. Supporting classifications with existing technology

There are many ways classifications could be handled in the main families of database models, however from the previous section it is clear that there are several requirements of the database to handle the range of classifications described above. In summary, for the simplest scenario of a single classification with no specific semantics, no overlap in categories nor traceability, they need to be able to represent and manipulate basic trees. For the more complex scenario of multiple overlapping classifications with specific semantics where traceability is required, they will require to represent and manipulate graphs with differing types of relationship with attributes to record their raison d'être.

This section examines the ability of relational, object-oriented, graph-based, and extended object-oriented models to support these requirements and discusses some specific techniques[2].

[2] Full details can be found in [37].

3.1. Representing single classifications

3.1.1. Trees and Graphs

The way classifications (trees or graphs) can be handled depends greatly on the structure of the database model. Relational models in general do not represent classifications easily, as they were originally designed for the manipulation of simple, flat data [10]. Extensions to the original relational model (extended model, in Third Manifesto [13]; object-relational models, e.g. Postgres [44], Oracle [30]) offer additional features such as extensible types or nesting that can be of use to describe more complex information. These models could handle graphs as relations, however these relations would have to play both the roles of nodes and edges in graphs, therefore their manipulation would need to be handled by user applications. Figure 7 shows a relation "Person" that could be related to its parents via the relation "Parents" playing the role of a relationship in a genealogy classification.

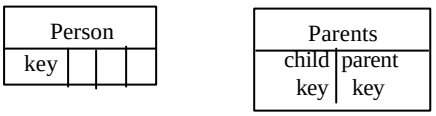

Figure 7: relation as relationship

Nested models could represent graphs via nesting, but this representation would still be simplistic as more complex graphs, e.g. weighted graphs, could not be captured. Finally, as all these options are too simple to accurately represent trees/graphs, they lead to complex processes/manipulations and possibly integrity constraint problems. These complex processes would have a negative impact on the efficiency of the overall system.

Object-oriented databases can support the definition of directed graphs (cyclic or not) using objects and references. However, only the simplest graphs (e.g. not weighted graphs) can be represented, as edges show only the existence of a link between two nodes (as a reference), without any additional information, which would be necessary for e.g. weighted graphs. An alternative approach would be the representation of graph edges by normal objects that user applications would recognise as edges. Figure 8 shows a class diagram where a class is used as relationship (with weight) to relate books to their class (category).

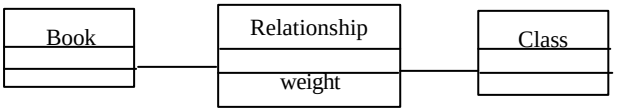

Figure 8: object as relationship

This has the advantage of providing a means to capture complex graphs such as weighted graphs. However, this would be limited in the sense that edge objects would not be recognised as relationships or references by the database system. Therefore, the insertion of an additional level of indirection would make user applications more complex and may lead to integrity problems (updating an object edge is more complicated than updating a reference). In addition, writing queries would be made harder by the additional level of indirection and the necessity to select these edge objects explicitly if they are required in the query result. Another approach could be to use the classification offered by some object-oriented models: classification of types by distinguishing types from classes [27] [22] [20]. However, changes to the classes may lead to important class reorganisation problems such as schema evolution (e.g. [4] [12]). They would also lead to very large schemas that could become unmanageable (a schema for a plant taxonomy flora would contain hundreds of thousands of classes).

In graph-based models, everything is represented by sets of nodes and edges that represent interaction between nodes. These models inherently support the description of trees/graphs, with various degrees of information: node-based models (e.g. TSIMMIS [9], Lore [25], GOOD [17]) have a weak representation of edges, and edge-based models (BDS95 [8]) limit the possibility to distinguish objects and interactions between objects. Models that support both kinds of features extensively (PROGRES [43], Telos [26], ConceptBase [21], [47], [46], Gram [3]), and in particular those that support nesting (Hyperlog [33], [49]), provide more freedom to choose the representation of information. By explicitly modelling nodes and edges of a graph, they allow the representation of any kind of graph, including weighted graphs or cyclic graphs.

Extended object-oriented models can represent graph structures as they are based on object-oriented models with first-class relationships. Their degree of support for graph structures varies: some support the definition of explicit but simple graphs (e.g. SORAC [16]); others support the definition of more complex graphs such as weighted graphs (e.g. GraphDB [19], ADAM [15], Albano [2]). In all cases, these graphs explicitly represent both nodes (objects for GraphDB and SORAC, or unary collections for OMS and Albano) and edges (specific relationship objects for GraphDB and SORAC, or binary/n-ary collections for OMS and Albano).

3.1.2. Semantics

Semantics of classifications is also an issue that is not handled well by most models. Relational models propose relations as the basic entity. These relations do not represent is-a, similarity, or part-of relationships in a system understandable manner, therefore relational

models would be unable to capture these kinds of classifications naturally. All classifications in a relational database would be generic classifications, which can lead to problems interpreting their semantics (once again captured by user applications).

Object-oriented models could only model two kinds of classifications: is-a (inheritance) and another generic kind of classification (reference). Is-a and generic classifications may not be appropriate to all classification schemes (e.g. part-of classifications as in the ICD), and no other specific kind of classification (e.g. part-of) may be defined.

Graph-based models generally only support one kind of relationship between nodes. Only a few support relationship classes and is-a classifications (e.g. Hyperlog [33]). If relationships are sub-classed, then it is possible to create classifications that are of the required type. The limitation of graph-based models is that they do not interpret the semantics of relationships. Therefore it is possible to describe generic classifications, but the system would not be able to interpret their meaning. For example is-a edges may be created in models that support the extension of edge types, but they would only be called "is-a" edges and inheritance rules would not be enforced.

By defining different kinds of relationships and using them to capture classifications, extended object-oriented models make it possible to define classifications that are not is-a or part-of classifications. This can be done by creating new kinds of relationships, with their semantics (e.g. as constraints or rules), and linking objects together. However, for the models that do not support semantic relationships, similarity and part-of classifications are impossible.

3.1.3. Traceability

Traceability offers the ability to record the motivation behind the building of classifications. Depending on the approach chosen for representing classifcations, traceability may be supported by relational models. For example, if relations are used to represent edges (instead of nodes), then these edges can contain information that can capture classification motivation, therefore traceability. If nesting is used, then no traceability information can be recorded.

Traceability is an unresolved issue in object-oriented and graph-based models. Indeed, if it is decided that traceability should be part of edges/relationships, only models that allow the definition of edges/relationships with weights may provide a solution. However, as the previous section explains, this approach is not practical with object-oriented models (where references cannot contain values, and normal objects used as relationships introduce problems) and graph-based models offer relationships of a too simple kind to handle attributes.

Some of the extended object-oriented models support attributes on relationships (Albano, GraphDB, ADAM), others allow the combinations of relationships as relationships (attributes) of relationships (e.g. OMS), therefore these classification relationships could also record the motivations for classifications as attributes of relationships forming a graph.

3.2. Representing multiple overlapping classifications

Identity is the main issue associated with multiple overlapping classifications. Indeed, the fact that several classifications may share elements means that it is important to be able to make a distinction between all the classifications involved. One feature offered by most relational systems, views [11], may be of interest as they would allow the filtering of relations according to specific criteria in order to present a partial view of the information to the user, e.g. a single or several classifications at a time. The idea is seductive but practical problems arise: the definition of views may be very complex, as a single classification contains a high number of different concepts (e.g. composite entities). The selection of all these concepts would require an inordinate number of queries (at least one for each table of interest), which would not only be hard to express (new relations may need to be created), but would be extremely expensive to compute.

In object-oriented databases, view mechanisms allow the definition of different appearances for objects and classes [42] [38] [6] [41]. The view mechanism can maintain a global schema [38] [5] [23] that contains all classifications, and extracts individual classifications to present them to the user. Figure 9 shows a mechanism by which views (top plane) are extracted from a global schema (bottom plane). Views are more flexible than schema-based approaches, as they can be created with the query language [14] or a view definition language [40], and some view mechanisms allow reorganisation and possibly automatic class integration in existing schemas and views [39]. However, the cost of creating and modifying views, even if they are materialised [24], may be too high to allow dynamic classifications. For example conflicts must be detected and resolved, and mistakes (especially in taxonomic work) might not allow this.

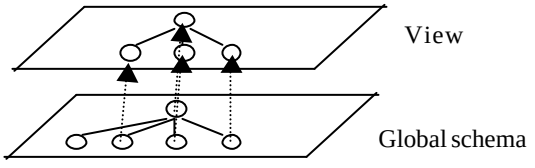

Figure 9: view as filtering mechanism

Views may also offer a way to express alternative classifications over graph-based models. View mechanisms have been proposed in the context of graph-based databases [49] [48] and semistructured databases [45] [1]. Graph views are essentially filtering mechanisms where sets of nodes and edges are selected. These proposals are all limited to specific aspects of graph databases: [49] only extracts views as sets of objects; [48] does not deal with updates and deletions; [45] only works with join-free queries and insert statements. Only [1] proposes a generic view mechanism that takes into account the specificity of graphs, and supports all operations. This view mechanism may allow the extraction of sub graphs from a general graph as classifications extracted from an overlapping larger classification.

Extended object-oriented models do not intrinsically support interconnected classifications. They only support the definition of unspecialised graphs. Additional mechanisms, at the model and/or at the query language level, would need to be developed in order to support multiple overlapping classifications. Unlike object-oriented and many graph-based models, no view mechanism is available for these approaches. In many cases, they are built from scratch in order to support uncommon features (e.g. GraphDB, SORAC, Albano) or are built on top of existing database systems that do not support views (e.g. ADAM). As a consequence, the representation of multiple overlapping classifications as views of a single larger classification is impossible.

4. Classification in context

The previous section has shown why existing technology fails to support multiple overlapping classifications and satisfy all the issues associated with their proper handling. Even mechanisms described in the literature for biological classifications (e.g. the *Materialization* relationship [31] and *power types* [28]) are too limited to support multiple overlapping classifications: they either work at class level (materialisation), which generate important classification reorganisation problems due to schema evolution, or do not support multiple overlapping classifications (materialisation, power types).

A new mechanism has been devised in the Prometheus project, which is described in this section. First, the technique for representing classifications is explained. This mechanism allows the representation of all types of classifications from single to multiple overlapping classifications. Then it is shown how the representation of multiple overlapping classifications does not impair the ability of the system to retain single points of views.

4.1. Relationships as classifiers

As the previous section explained, the model that offers the best support for classification representation is the extended object-oriented model. It combines the high level approach of object-oriented models with the decomposition, low level approach offered by graph-based models. This allows extended object-oriented models to capture some forms of classification. However, the previous section has also shown that this model fails to capture multiple overlapping classifications properly. The approach that has been taken for the Prometheus project is the use of such a model combined with additional mechanisms. We use relationships with the equivalent of weights (as in weighted graphs) to describe classifications.

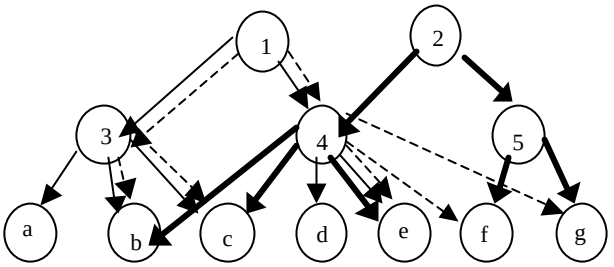

Figure 10: Multiple overlapping classification example

Relationships effectively act as classifiers (or the classifying mechanism). The action of creating such a relationship between two objects implies that these objects are classified. Furthermore, these relationships are the only objects in the system that are aware of the classifications and they contain all the necessary information to distinguish them from each other. They therefore support the independence of classifications and data. Figure 10 shows how the use of relationships as classifiers allows the description of multiple overlapping classifications. Each of these classifications represents a specific opinion, i.e. the context in which the specimens are classified. Figure 10 shows three distinct classifications: a dashed line classification, a thin line classification, and a thick line classification. In taxonomy, these distinct classifications would have been published by distinct authors and the publication information would replace the type of arrow in this example. The leaf nodes in these classifications could be for example books or plant specimens. The other nodes can be book subjects or taxa that are used to classify the leaf nodes.

4.2. Traceability

Traceability is handled by the relationships that act as classifiers. Traceability information is part of the classification information, therefore this is the right place to hold it. As relationships can have weights (captured by

attributes), possibly also as part of the context (e.g. if following paths that only contain certain values is of interest), some of their attributes may be used to record decisions. For example, the edge between nodes 4 and b Figure 10 may have been created because 4 and b exhibit a specific property. This decision can be captured by an attribute of that relationship (with something as simple as free text if necessary, or more complex object structures).

4.3. Semantics

Semantics are provided by the fact that all relationships can participate in the classification description and that the model offers extension of relationships by sub-classing and description of behaviour and constraints. If part-of classifications are to be described, then aggregation relationships can be used and their semantics interpreted by the system. If other kinds of classifications are to be represented (e.g. similarity), new relationships, with specific semantics, can be created and used.

4.4. Multiple overlapping classifications

It can be seen in Figure 10 that the different classifications have elements in common: node 3 appears in the thin line classification and in the dashed line classification; node 4 appears in all three classifications. On the contrary, node 5 only appears in the thick line classification. Likewise, the leaf nodes can appear in one classification (node a), in two classifications (node b), or in all three (node e).

As Figure 10 shows, identity of distinct classifications can be handled through the type of the relationships that are used to describe them. Sub-classing of relationships (as first-class objects) allows for example the creation of specific relationships for each classification to be represented. It is also possible to manage this identity using attributes of relationships: specific values may represent each classification. In any case, distinct classifications are clearly identified and this distinction does not impair querying, as querying attributes and using types is inherent in object-oriented query languages.

4.5. Classifying in context

This new approach allows the generic classification of entities by context. By "context", one can understand "anything that uniquely identifies a view". In plant taxonomy, this can be a taxonomist, a publication, or a combination of both. For example, one taxonomist's view on the world is a context and in that context a set of specimens is classified in a certain way. Concurrently, another taxonomist's view of the world represents another context where the same specimens (or any other set of specimens) are classified differently. The overall graph that is stored in the database represents a view of taxonomy out of context, or within all contexts concurrently. This view, although it is the most complete because it contains all existing information, does not suit some classification work (e.g. taxonomy work), as users tend to work in one particular context or in relation to a limited set of contexts for comparison purposes. By representing classification information on the hierarchies that constitute that graph, Prometheus captures single contexts that can be extracted as necessary.

Because the distinct hierarchies created in different contexts overlap (in terms of categories and classified concepts), the representation of all contexts in a single graph makes possible the comparison of classifications defined in different contexts and provides the ability to switch between contexts in order to gain knowledge. Indeed, by following relationships with specific values (e.g. publication information), it is possible to follow a path of a specific graph. But by switching between these values, it is possible to compare and navigate within and amongst classifications. For example in Figure 10, it is possible to compare nodes 3 and 4 and thereby to realise that they have some leaf nodes in common. This can give new insight into the data (e.g. in plant taxonomy when two groups partially contain the same specimens, they are partial synonyms). It is also possible to contrast the different meanings of node 4 according to the different classifications: it contains nodes d and e in the thin line classification, nodes e, f, and g in the dashed line classification, and nodes b, c, and e in the thick line classification.

5. Conclusion

This paper has presented the concept of classification as ranging from single taxonomy to multiple overlapping classifications, appearing in many areas of science. The latter case represents the multiple overlapping classification of objects or classes in separate but overlapping classifications. The features identified as necessary for handling these classifications include trees/graphs, traceability, semantics of classifications, independence of classification and data, and identity of classifications.

Common database models have been investigated for their support of multiple overlapping classification regarding the requirements expressed, and we have concluded that none offers full supports for the features outlined, but many provide a part of a satisfying solution. A new method of capturing multiple overlapping classifications has therefore been devised where context, i.e. what identifies one classification from another, plays a central role. The approach uses an extended object-oriented model to capture classification information and links, so that classes and classified objects can be related

by classification information but stay independent from classifications.

The technique presented here has been implemented and tested in a plant taxonomy database system, and has been shown to be effective in handling multiple classifications and their associated processes. However this approach is applicable to all domains where contexts or multifaceted objects exist. For example, context is important for ontologies, as pointed out by Priss [34], and is often ignored. An approach such as that proposed here could be applied to ontology systems in order to introduce the concept of context.

6. References

[1] S. Abiteboul, J. M. Hugh, M. Rys, V. Vassalos, and J. Wiener, "Incremental maintenance for materialized views over semistructured data," presented at VLDB'98, Proceedings of 24rd International Conference on Very Large Data Bases, New York City, New York, USA, 1998.

[2] A. Albano, G. Ghelli, and R. Orsini, "A Relationship Mechanism for a Strongly Typ ed Object-Oriented Database Programming Language," presented at Proceedings of the seventeenth international conference on very large data bases, Barcelona, Spain, 1991.

[3] B. Amann and M. Scholl, "Gram: A Graph Data Model and Query Language," INRIA, Le Chesnay, France Verso report number 046 (ECHT), 1992.

[4] J. Banerjee, H. Chou, J. Garza, W. Kim, D. Woelk, and N. Ballou, "Data model issues for object-oriented applications," *ACM Transactions on Office Information Systems*, vol. 5, pp. pp 3-26, 1987.

[5] Z. Bellahsene, "View Mechanism for Schema Evolution in Object-Oriented DBMS," presented at 14th British National Conferenc on Databases, BNCOD 14, Edinburgh, Scotland, 1996.

[6] Z. Bellahsene, "Updating Virtual Complex Objects," presented at OOIS'97, 1997 International Conference on Object Oriented Information Systems, Brisbane, Australia, 1997.

[7] G. C. Bowker and S. L. Star, *Sorting things out, classification and its consequences*: Massachusetts Institute of Technology, 1999.

[8] P. Buneman, S. Davidson, and D. Suciu, "Programming Constructs for Unstructured Data," presented at DBPL-5 Proceedings of the Workshop on Database Programming Languages, Gubbio, Umbria, Italy, 1995.

[9] S. Chawathe, H. Garcia-Molina, J. Hammer, K. Ireland, Y. Papakonstantinou, J. U llman, and J. Widom, "The TSIMMIS Project: Integration of Heterogeneous Information Sources," presented at Proceedings of IPSJ Conference, Tokyo, Japan, 1994.

[10] E. F. Codd, "A Relational Model of Data for Large Shared Data Banks," *Communications of the ACM (CACM)*, vol. 13, pp. 377-387, 1970.

[11] S. J. Connan and G. A. M. Otten, *SQL - The Standard Handbook*: McGraw-Hill, 1992.

[12] V. Crestana-Taube and E. A. Rundensteiner, "Schema Removal Issues for Transparent Schema Evolution," presented at Sixth International Workshop on Research Issues on Data Engineering, Interoperability of Non traditional Database Systems, RIDE'96, IEEE, New Orleans, Louisiana, 1996.

[13] H. Darwen and C. J. Date, "The Third Manifesto," *SIGMOD Record 24*, vol. 24, pp. 39-49, 1995.

[14] S. Deßloch, T. Härder, F.-J. Leick, N. M. Mattos, C. Laasch, C. Rich, M. Scholl, and H.-J. Schek, "COCOON and FRISYS - a comparison -," in *Objektbanken für Experten, Informatik Aktuell*, T. H. R. Bayer, P.C. Lockemann, Ed.: Springer, 1992, pp. pp 179-196.

[15] O. Díaz and P. M. D. Gray, "Semantic-rich User-defined Relationship as a Main Constructor in Object Oriented Database," presented at Object-Oriented Databases: Analysis, Design & Construction (DS-4), Proceedings of the IFIP TC2/WG 2.6 Working Conference on Object-Oriented Databases: Analysis, Design & Construction, Windermere, UK, 1990.

[16] M. Doherty, J. Peckham, and V. F. Wolfe, "Implementing Relationships and Constraints in an Object-Oriented Database Using a Monitor Construct," in *Rules in Database Systems*, M. H. W. Norman Paton, Ed. Edinburgh: Springer-Verlag, 1993, pp. 347-363.

[17] M. Gemis, J. Paredaens, I. Thyssens, and J. V. d. Bussche, "GOOD, A graph-Oriented Database System," *Proceedings of SIGMOD, SIGMOD Record*, vol. 22, pp. 505--510, 1993.

[18] W. Greuter, F. R. Barrie, H. M. Burdet, W. G. Chaloner, V. Demoulin, D. L. Hawksworth, P. M. Jørgensen, D. H. Nicolson, P. C. Silva, P. Trehane, and J. McNe, *International code of botanical nomenclature (Tokyo Code)*, vol. 131: Koeltz Scientific Books, 1994.

[19] R. H. Güting, "GraphDB: Modeling and Querying Graphs in Databases," presented at Proc 20th Int. Conf. on Very Large Databases, Santiago, Chile, 1994.

[20] N. Hori, M. Yoshikawa, and S. Uemura, "ASKA: An Object-Oriented Data Model with Multiple Hierarchies and Multiple Object-Perspectives," presented at 6th Int. Conf. and Workshop on Database and Expert Systems Applications (DEXA'95) - Workshop Proceedings, London, UK, 1995.

[21] M. Jarke, R. Gallersdörfer, M. A. Jeusfeld, M. Staudt, and S. Eherer, "ConceptBase - a deductive object base for meta data management.," *Journal of Intelligent Information Systems. Special Issue on Advances in Deductive Object-Oriented Databases*, vol. 4, pp. 167-192, 1995.

[22] J. Joseph, S. Thatte, C. Thompson, and D. Wells, "Object-Oriented Databases: Design and Implementation," *Proceedings of the IEEE*, vol. 79, pp. 42-64, 1991.

[23] W. Kim and W. Kelley, "On View Support in Object-Oriented Database Systems," in *Modern database systems: The Object Model, Interoperability, and Beyond*, W. Kim, Ed. New York: Addison-Wesley Publishing Company, 1995, pp. 108-129.

[24] H. A. Kuno and E. A. Rundensteiner, "Materialized Object-Oriented Views in MultiView," presented at

Fifth International Workshop on Research Issues on Data Engineering: Distributed Object Management (RIDE-DOM'95), IEEE, 1995, Taipei, Taiwan, Taipei, Taiwan, 1995.

[25] J. McHugh, S. Abiteboul, R. Goldman, D. Quass, and J. Widom, "Lore: A Database Management System for Semistructured Data," *SIGMOD Record*, vol. 26, pp. 54-66, 1997.

[26] J. Mylopoulos, A. Borgida, M. Jarke, and M. Koubarakis, "Telos: Representing Knowledge About Information Systems," *ACM Transactions on Information Systems*, vol. 8, pp. 325-362, 1990.

[27] M. C. Norrie, "Distinguishing Typing and Classification in Object Data Models," in *Information Modelling and Knowledge Bases VI*, vol. VI, Chapter 25, H. Kangassalo, H. Jaakola, S. Oshuga, and B. Wangler, Eds.: IOS, 1995.

[28] J. Odell, "Power types," *Journal of Object-Oriented Programming*, vol. 7, pp. 8-12, 1994.

[29] J. Odell, "Six different kinds of composition," *Journal of Object-Oriented Programming*, vol. 6, pp. 10-15, 1994.

[30] Oracle, "Oracle, http:///www.oracle.com," 2001.

[31] A. Pirotte, E. Zimányi, D. Massart, and T. Yakusheva, "Materialization: a powerfull and ubiquitous abstraction pattern," presented at Very Lage Data Bases (VLDB'94), Santiago, Chile, 1994.

[32] M. K. a. A. Pirotte, "An aggregation model and its C++ implementation," presented at 4th Int. Conf. on Object-Oriented Information Systems, OOIS'97, Brisbane, Australia, 1997.

[33] A. Poulovassilis and S. Hild, "Hyperlog: a graph-based system for database browsing, querying and update," *To appear in IEEE Knowledge and Data Engineering*, 1998.

[34] U. Priss, "Ontologies and Context," presented at 12th Midwest Artificial Intelligence and Cognitive Science Conference, Miami University, Oxford, OH, USA, 2001.

[35] Prometheus, "Prometheus project web page," 1998.

[36] M. R. Pullan, M. F. Watson, J. B. Kennedy, C. Raguenaud, and R. Hyam, "The Prometheus Taxonomic Model: a practical approach to representing multiple taxonomies," *Taxon*, vol. 49, pp. 55-75, 2000.

[37] C. Raguenaud, "Managing complex taxonomic data in an object-oriented database," in *School of Computing*. Edinburgh: Napier University, 2002.

[38] E. A. Rundensteiner, "MultiView: A Methodology for Supporting Multiple View in Object-Oriented Databases," presented at 18th International Conference on Very Large Data Bases, Vancouver, Canada, 1992.

[39] E. A. Rundensteiner, "A Classification Algorithm For Supporting Object-Oriented Views," presented at Proceedings of the Third International Conference on Information and Knowledge Management (CIKM'94), Gaithersburg, Maryland, 1994.

[40] E. A. Rundensteiner and L. Bic, "Automatic View Schema Generation in Object-Oriented Databases," Department of Information and Computer Science, University of California, Irvine 92-15, 01/92 1992.

[41] C. S. d. Santos, S. Abiteboul, and C. Delobel, "Virtual Schemas and Bases," presented at Advances in Database Technology - EDBT'94. 4th International Conference on Extending Database Technology, Cambridge, United Kingdom, 1994.

[42] M. E. S. a. H.-J. Schek, "Supporting Views in Object-Oriented Databases," *IEEE Database Engineering Bulletin, Special Issue on Foundations of Object-Oriented Database Systems*, vol. 14, pp. 43-47, 1991.

[43] A. Schürr, A. J. Winter, and A. Zündorf, "PROGRES: Language and Environment," in *Handbook on Graph Grammars: Applications*, vol. 2, G. Rozenberg, Ed., Singapur: World Scientific ed, 1998.

[44] M. Stonebraker, A. Jhingran, J. Goh, and S. Potamianos, "On rules, procedures, caching and views in database systems," presented at ACM SIGMOD International Conference on Management of Data, Atlantic City, NJ, USA, 1990.

[45] D. Suciu, "Query Decomposition and View Maintenance for Query Languages for Unstructured Data," presented at VLDB'96, Proceedings of 22th International Conference on Very Large Data Bases, Mumbai (Bombay), India, 1996.

[46] F. W. Tompa, "A data model for flexible hypertext database systems," *ACM Transactions on Information Systems*, vol. 7, pp. 85-100, 1989.

[47] C. Watters and M. A. Shepherd, "A Transient Hypergraph-Based Model for Data Access," *ACM Transactions on Information Systems*, vol. 8, pp. 77-102, 1990.

[48] P. T. Wood, "Graph Views and Recursive Query Languages," presented at BNCOD 8, University of York, UK, 1990.

[49] Y. Zhuge and H. Garcia-Molina, "Graph Structured Views and Their Incremental Maintenance," presented at Proceedings of the Fourteenth International Conference on Data Engineering, Orlando, Florida, USA, 1998.