Identifying a Forest Hierarchy in an OODB Specialization Hierarchy Satisfying Disciplined Modeling

Yehoshua Perl, James Geller, and Huanying(Helen) Gu *
Institute for Integrated Systems
Department of Computer and Information Sciences
& Center for Manufacturing Systems
New Jersey Institute of Technology
Newark, NJ 07102
geller@homer.njit.edu
phone: (201) 596-3383

Abstract

Our work is motivated by the desire to develop methods to comprehend large vocabularies and large schemas of Object-Oriented Databases. The ability of a user of a database participating in a federated system to retrieve information from the other database systems will be greatly enhanced by acquiring a better comprehension of these systems. We are trying to develop both a theoretical paradigm and a methodology to analyze existing large schemas. Our approach to achieve comprehension is based on combining two concepts: informational thinning (i.e. concentration on the specialization hierarchy of the schema) and partitioning. In this paper we present a new technique for modeling which is called disciplined modeling. Based on the rules of disciplined modeling we develop a theoretical paradigm to support the existence of a meaningful forest hierarchy within the specialization hierarchy. Such a hierarchy functions as a skeleton of the schema and supports comprehension and partitioning efforts.

1 Motivation

One of the problems in cooperative information systems is that the user of each system is familiar with his own applications, databases and terminologies, but he is much less familiar with those of the other systems. Therefore, the cooperation suffers from the lack of comprehension of the schemas and terminologies used in the other federated systems. Our work is motivated by the desire to comprehend large vocabularies and schemas of Object-Oriented Databases (OODB) and to develop methods to make them understandable for users. We are developing a theoretical framework and a methodology which will enable the design of tools to aid comprehension of existing OODB systems. Graphical presentation is a very important tool to support schema comprehension. Thus, we assume that schemas and vocabularies are represented with a graphical schema language and manipulated with a graphical editor. Our approach to achieve comprehension of graphical schemas is based on two concepts: *informational thinning* (i.e. concentration on the specialization hierarchy) and *partitioning*.

In this paper we present a new technique for modeling called *disciplined modeling*. Based on the rules of *disciplined modeling* we develop a theoretical paradigm to support the existence of a meaningful forest within a specialization hierarchy. This forest represents a partitioning of the specialization hierarchy into trees and functions as a skeleton of the schema that supports comprehension efforts. In [1] we will present a methodology based on our paradigm for finding a forest hierarchy for a given schema.

This research is part of our current OOHVR (Object-Oriented Healthcare Vocabulary Repository; pronounce: "over") project and continues our previous work on OODB schemas [2, 3]. In order to understand the pressing need for this research, we will now describe the medical vocabulary that we are using. The cur-

*This research was (partially) done under a cooperative agreement between the National Institute of Standards and Technology Advanced Technology Program (under the HIIT contract, number 70NANB5H1011) and the Healthcare Open Systems and Trials, Inc consortium. This research was partially supported by a grant from the Center for Manufacturing Systems at NJIT.

0-8186-7505-5/96 $5.00 © 1996 IEEE

rent MED (Medical Entity Dictionary) built at CPMC[1] [4, 5] is based on a semantic network model. In December of 1995 the MED contained 42886 concepts, 573939 attributes, 54359 subclass relationships, and a total of 255160 relationships, and the MED keeps growing. The vocabulary is built around a hierarchy of specialization relationships known as "IS A" or SUBCLASS. The hierarchy allows multiple inheritance, that is, an object is a specialized object of several other objects simultaneously. Although the dictionary contains other kinds of relationships as well, this hierarchy serves as the skeleton of the vocabulary, as it provides defining information and supports inheritance of properties. In [1] we will demonstrate our methodology applied to a subschema of the MED.

Our methods are applicable to vocabularies, Object-Oriented schemas, semantic networks, Entity Relationship models, semantic data models. It may be applicable to part relationship hierarchies [6, 7]. For example, the MED contains a part hierarchy of about 2500 part relationships.

2 Schema complexity and schema partitioning

2.1 Schema complexity

To define what it means for a schema to be large and complex, we equate the *size* of the schema with the number of classes. Our experience is that comprehension difficulties of a schema stem more from the number of relationships than from the number of classes. We define the **complexity**, **c**, of a schema as the ratio of the number of the relationships between classes, and the number of classes. For two schemas of equal size, a more complex schema is more difficult to comprehend.

In our previous work [8, 9, 10] we used a large OODB schema describing a university. In this paper, we will demonstrate our techniques on a schema from the university environment, as most readers have better knowledge about professors and chairpersons, than about medical terms such as nucleotide sequences and antibiotic sensitivity panels.

One method that has been widely used to make schemas comprehensible is to use a graphical representation. We note, e.g., the popularity of the Entity Relationship model and its graphical representation [11] and graphical case tools such as OMT [12] and ROSE [13].

Following this, we present (Figure 1) a subschema of the university OODB schema concentrating on the academic aspects of the university which contains 36 classes, about a third of the whole schema. (The schema was simplified by omissions.)

The schema is presented using our graphical OODB language OODINI [14]. In OODINI a rectangle represents a class, and a double line rectangle represents a set class which shares a corner with the rectangle of its member class. (See e.g. the class **section** and its set class **crsections**.) A thin arrow represents a relationship between two classes, and a double thin arrow stands for a multivalued relationship. In OODINI we differentiate between two SUBCLASS relationships. The *category of* relationship using a bold line and the *role of* relationship with a bold line composed of dots and dashes (Section 3). An attribute is presented by an ellipse connected to its class with a thin line. Only one attribute "teach Eval" of the class **instructor** is shown in Figure 1.

The schema contains *74* relationships and thus its complexity ratio **c** = *74/36 = 2.06*. A user needs a substantial effort to obtain an orientation in this schema. This is rather distressing, because this is only a simplified subschema.

2.2 Informational thinning

We will now talk about two approaches towards making a complex graphical display understandable. *Informational thinning* tries to eliminate information from the whole schema by prioritizing various kinds of properties of the classes and displaying only high priority kinds of properties. E.g., the OODINI graphical editor permits three levels of display: 1) Full schema of classes, attributes, and all relationships (no *informational thinning*); 2) Classes and all relationships, but no attributes (partial thinning); 3) Classes and subclass relationships only, but no attributes and no user-defined relationships (full thinning). In Figure 2 we display only the classes and the subclass relationships of the schema of Figure 1. Note that for Figure 2 **c** = *26/36 = 0.72*. We will refer to display level 3) (Figure 2) as *hierarchical (sub)schema of a schema*. This subschema has the same size of the original schema but a lower complexity and it is easier to comprehend. Furthermore, it is easier to comprehend, because the designer does not have to label the subclass relationship, while for the omitted user defined relationships labels were necessary to specify their semantics.

For large schemas even the hierarchical subschema may be difficult to comprehend due to its size and multiple inheritance. This is certainly true for the MED. The **complexity** of the hierarchical subschema of the MED **c** = *54359/42886 = 1.27* compared to the orig-

[1] Columbia Presbyterian Medical Center

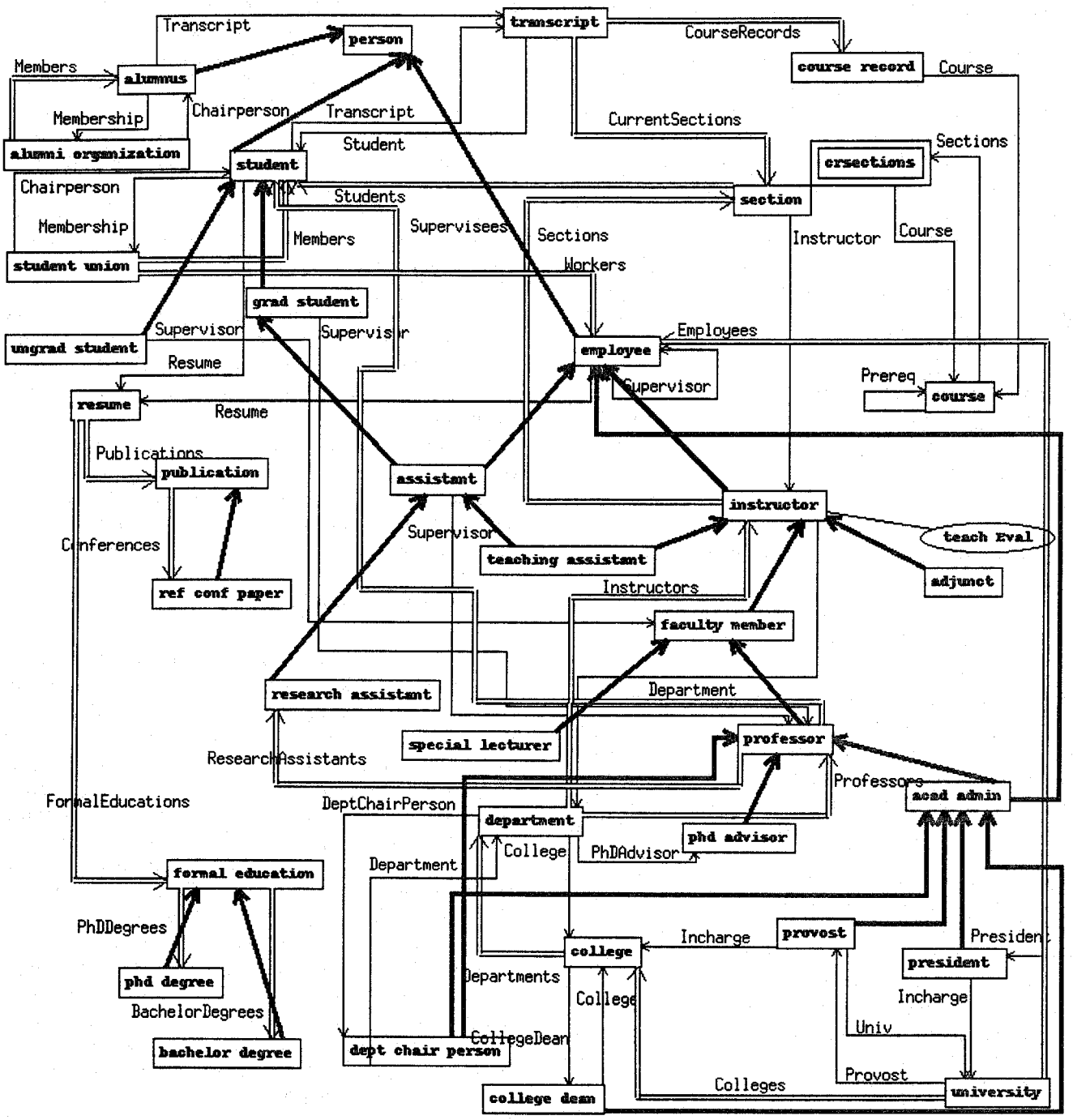

Figure 1. A Subschema of a University Database

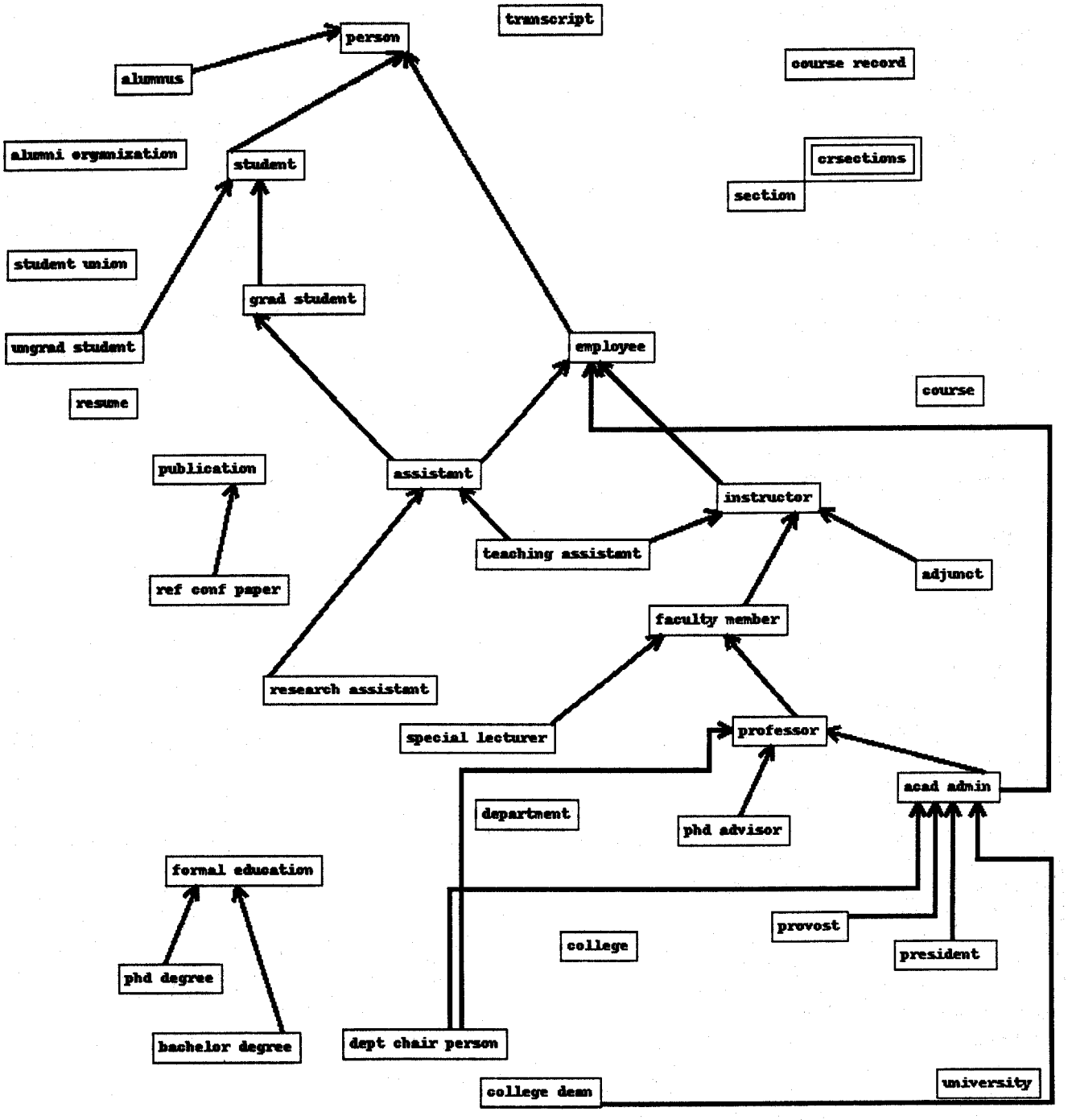

Figure 2. A Subschema of University Database at display level 3)

inal schema of the MED where $\mathbf{c} = 255160/42886 = 5.95$.

The subclass relationship, by definition, forms a directed acyclic graph (DAG). In graph theory, a hierarchical schema has a forest structure if no class has more than one superclass. If a forest hierarchical subschema is connected, then it forms a spanning tree of the DAG. It is generally considered to be easier to comprehend a forest hierarchical schema than a DAG of the same size, because upward paths are not branching in a forest.

2.3 Schema partitioning

A second approach to simplify the comprehension of a complex large schema is partitioning it into smaller subschemas. This applies to general and hierarchical schemas alike. From the technical side, only a limited size subschema can be displayed on a screen. From the conceptual side, human comprehension capacity is limited. Hence, we are faced with the task to partition a large schema into smaller subschemas. In the partitioning we have two purposes:

1. To identify small subschemas which comprise logical units of the original schema.

2. To generate a minimal number of subschemas each of which fits on a computer screen by itself, and which together comprise the complete schema.

The need for a logical partitioning of a schema introduces a vicious cycle, as one needs to comprehend the schema to partition it logically. Nevertheless, partitioning into logical subschemas tends to minimize the number of relationships between different subschemas. Unfortunately, the problem of partitioning a schema according to the above or similar criteria is NP-complete, that is, no efficient algorithm is known for it, and it is conjectured that no such algorithm exists [15].

A possible line of action is to combine informational thinning and partitioning, by trying to partition the hierarchical subschema and then use this partition to impose a partition on the original schema. Obviously, this partitioning problem is much simpler than the original problem since the schema has less relationships. However, unless the hierarchical subschema has a forest structure, the partitioning problem in general is still NP-complete. On the other hand, if it has a forest structure then there exist efficient algorithms for various partitioning criteria [16, 17, 18, 19, 20]. In this paper, we will show that in a hierarchical subschema of a general schema, it is possible to identify a forest hierarchical subschema, the semantics of which helps to support comprehension.

3 The rules of disciplined modeling

3.1 The category of and role of specialization relationships

In order to identify a meaningful forest subschema of a hierarchical schema we shall look into the nature of the specialization relationship. In previous research we [21] and others [22] have identified two different kinds of SUBCLASS relationships, namely, *category of* and *role of*. According to our definition, *category of* is a specialization relationship used for refinement in case that the superclass and the subclass are in the same context. On the other hand, *role of* is used when the superclass and the subclass are in different contexts.

How does a designer of an OODB schema determine whether a given SUBCLASS relationship is *category of* or *role of* ? This depends on whether the two connected classes are in the same context or not. For example, a student is a person, and a graduate student is a student. However, intuition tells us that information about the student and the graduate student are both in the same context, while person is in a different context. Hence, the graduate student is *category of* student, which in turn is *role of* person. However, this determination is not always so easy. In spite of extensive research, e.g., [23, 24, 25, 26, 27, 28, 29] there is still no widely accepted definition of context. Building a gigantic knowledge base in the CYC project [30] was found doomed to failure were contexts not introduced as a structuring mechanism. Work following this line [23, 25, 27] assumes that a context is a first class object used to parameterize axiom schemata. However, no clarity about the nature of contexts themselves is gained by this approach. As a recent workshop on context in Natural Language Processing showed [26] researchers currently agree that they disagree on what contexts are. Our approach is that we are not trying to define the notion of context. Rather we are making the pretheoretical (axiomatic) assumption that contexts exist in human thinking and we are trying to identify them.

In this paper, we provide a theoretical paradigm for the existence of such assignments of classes to contexts which results in a forest subschema of a DAG, the semantics of which supports comprehension of the schema. In [1] we will introduce a methodology for finding such a forest.

In order to ensure that a forest hierarchical subschema can be identified, the assignment of classes to contexts must always satisfy three rules of *disciplined modeling*. As we shall show, a schema designer using *disciplined modeling* can still model the same situations

as before. Only few modifications are required in the modeling, so that the rules of *disciplined modeling* are satisfied.

3.2 Contexts as equivalence relations

First we define the mathematical relation *equicontext*, or "in the same context," between classes. A pair of two classes belongs to the equicontext relation if both classes belong to the same context.

RULE 1: The equicontext relation between classes is an equivalence relation, that is, it is satisfying the 3 conditions of an equivalence relation: reflexivity, symmetry and transitivity.

An equivalence relation partitions the elements of a set into disjoint subsets, such that every two elements of the same subset are related and no two elements of different subsets are related. Hence, **RULE 1** implies **RULE 1'**.

RULE 1': The classes of a schema are partitioned by the equicontext relation into disjoint contexts.

RULE 1' will force the designer into explicit specification of the contexts in his schema and lead him to resolve some ambiguous situations in a systematic way. We do not claim to have a unique way of assigning classes to contexts. As we are dealing with a problem of data modeling, there are usually different ways to model the same real world environment. We further do not claim that contexts are naturally disjoint. To the contrary, in many applications, initial contexts may overlap. However, *disciplined modeling* forces the modeler to design disjoint contexts.

3.3 A category of refinement is exclusive

The *category of* relationship is used when we need to refine the concept represented by the superclass when both the superclass and the subclass are all in the same context. This means that instances of the superclass are divided into being also instances of the different *category of* subclasses according to some distinction. In *disciplined modeling*, we further require from such a categorization that the refinement will be into mutually exclusive concepts.

RULE 2: Two classes which are *category of* specializations of a superclass cannot both contain an instance representing the same real world object.

This means that an instance that belongs to the extent of a superclass cannot belong simultaneously to the extent of more than one subclass. In other words, the real world objects corresponding to the instances of the different subclasses form disjoint sets.

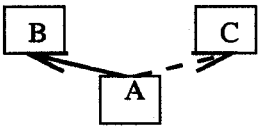

Figure 3. Resolution of CASE 1

Consider how to guarantee **RULE 2** in *disciplined modeling*. Let a class A be a subclass of two classes B and C. By **RULE 2** it cannot be that both such subclass relationships are *category of*. Thus we need to give the disciplined modeler guidelines how to deal with the modeling of such a situation.

CASE 1: When examining the application closely the modeler realizes that of the two super classes of class A one, say B, is a primary superclass while the other, say C, is a secondary superclass, or in other words, one superclass is considered more important than the other. Here, the modeler will include A in the context of B while C will belong to another context. Hence, the relationship of A to B will be *category of* while the relationship of A to C will be *role of* (Figure 3). Since the primary superclass is not always easy to determine, we provide the following further guidelines.

CASE 1.1: One superclass describes the essence or the definition of the subclass, while the other superclass describes the functionality or usage of the subclass. Here the definitional superclass is primary.

CASE 1.2: Both superclasses are definitional, however, it is possible to select the primary by a linguistic analysis of the name of the subclass. For example, when the subclass name consists of an adjective and a noun characterizing the two superclasses then the noun defines the primary superclass. As another example, when both concepts are nouns then the second noun is considered primary. Here we follow the structure of a noun phrase consisting of a modifier noun, followed by a head noun.

CASE 2: Both superclasses are definitional with indistinguishable importance. In this case we have no reason to prefer one over the other. By the rules of *disciplined modeling*, each choice of context will disassociate this class from the other context. The conclusion is to create a new context for this concept, an intersection concept of both superclasses. Let us demonstrate these guidelines by some examples.

Example 1: Figure 4 describes classifications of dog, cat, cheetah and wolf according to their families, feline or canine, and according to being wild or domesticated. Thus, each of the four kinds has two superclasses. We apply **CASE 1.1** where the animal family gives defi-

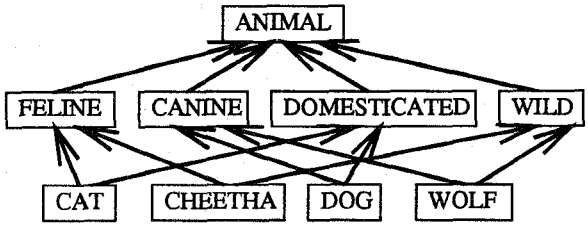

Figure 4. Multiple inheritance CASE 1.1

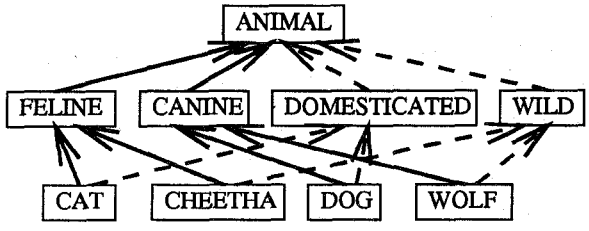

Figure 5. Multiple inheritance CASE 1.1 resolved

nitional information while being wild or domesticateed is a functional description. The results of the analysis appear in Figure 5 where no class has two *category of* superclasses. Hence, **RULE 2** is satisfied.

Example 2: A class person has two subclasses **Quaker** and **Republican**. These classes in turn have a joint subclass **Republican–Quaker**. This example is known as "Nixon Diamond" [31, 32] (Figure 6). By **CASE 1.2** the primary superclass for **Republican–Quaker** is **Quaker**. There are three contexts in this example. A personal context containing the class **person**, a religious context containing **Quaker** and **Republican-Quaker**, and a political context containing **Republican**. Thus, we define three of the specializations as *role of* relationships (Figure 7) and **RULE 2** is satisfied.

Example 3 is taken from the MED: The class **Polysporin topical oint 30 gm** represents a drug. It has a superclass **Bacitracin / polymyxin b combination preparations** which in turn has two

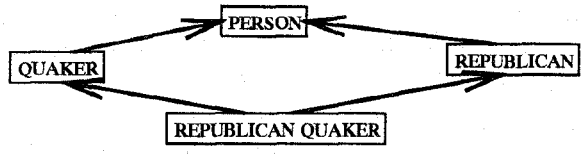

Figure 6. Multiple inheritance CASE 1.2: The Nixon Diamond

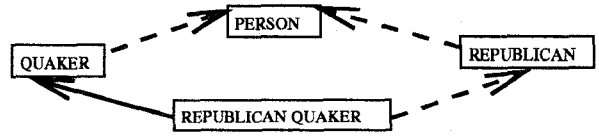

Figure 7. Multiple inheritance CASE 1.2: The Nixon Diamond resolved

structural superclasses **Bacitracin preparation** and **Polymyxin b preparation** both of which are subclasses of **Miscellaneous antibiotics** (Figure 8). The slash syntax of **Bacitracin / polymyxin b combination preparations** means that it is a mixture without identifying a priority of one ingredient (superclass) over the other. According to **CASE 2**, **Bacitracin / polymyxin b combination preparations** is therefore in its own context which is different from the two contexts of its superclasses. Hence, **Bacitracin / polymyxin b combination preparations** is *role of* its two superclasses. The classes **Polysporin topical oint 30 gm** and **Polysporin opth oint 3.5 gm** are *category of* the class **Bacitracin / polymyxin b combination preparations** since it is their structural superclass (Figure 9).

3.4 Contexts need roots

Let us now extend Example 2, to several religions and two political orientations (Figure 10). We identify three different contexts: personal, religious and political. This example lead us to introduce the third rule of *disciplined modeling.*

RULE 3: For each context there exists one class which is the *major* (or defining) class for this context such that every class in this context is a descendent of this class.

In other words, each context has one class which is a "root" for it (i.e., there is a directed path from each class of the context to this class) with regards to a directed graph of *category of* relationships. Note that we use here the notion of a directed tree where all the directions are towards the "root". I.e., in graph theory terms the root is a sink.

Figure 10 does not satisfy **RULE 3**. Therefore, we have to group together the two contexts, the religious and the political, by introducing two new classes as *role of* **person**. The various religious orientations are *category of* the class **religious person** and the political orientations become *category of* the class **political person** (Figure 11).

Finally, the classes on the lower level, **Republican–Quaker** and **Democrat-Catholic** are handled as in

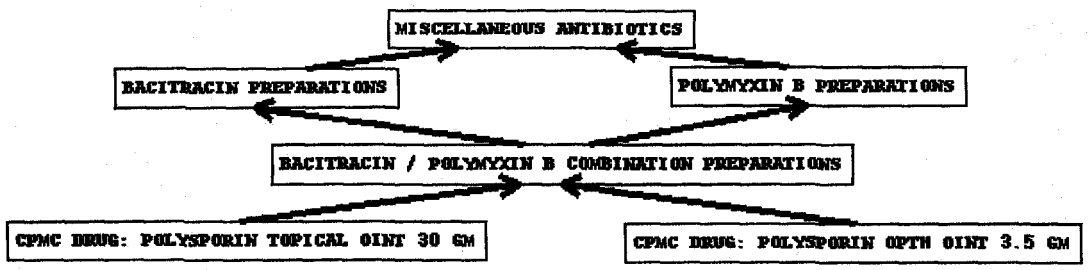

Figure 8. Multiple Inheritance CASE 2

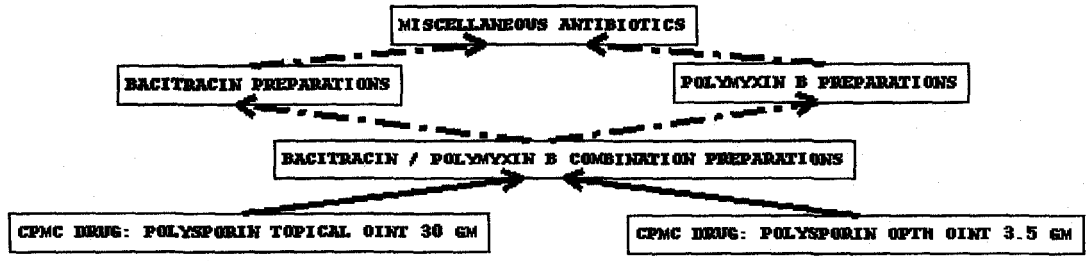

Figure 9. Multiple Inheritance CASE 2 resolved

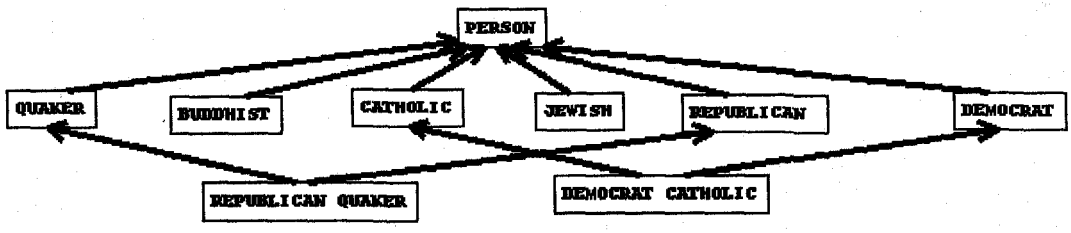

Figure 10. Extended Nixon Diamond

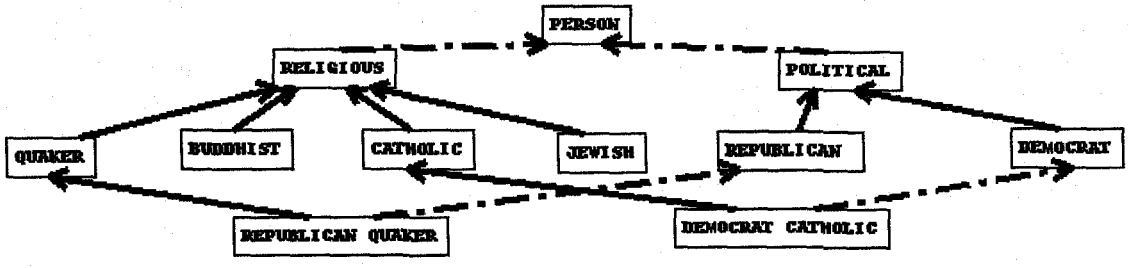

Figure 11. Extended Nixon Diamond resolved with RULE 3

Figure 7, i.e., they are *category of* the religious orientation and *role of* the political orientation. In this case *disciplined modeling* will permit moving some common properties of the various religious (political) classes into the newly created class **religious person (political person)** so they are inherited to each of the religious (political) classes. This shows that adding the extra layer in the hierarchy leads to better modeling. That is, we encounter a tradeoff of adding two new classes versus removing some explicit specifications of properties from the other classes which become children of the new classes.

Alternatively, one can satisfy **RULE 3** by keeping the original modeling with smaller contexts, using each of the religious and political denominations as roots. These two alternative modeling choices lead to another general issue. There are potentially different nesting levels in identifying contexts. In the above example, we could have either one religious context or divide it into several more detailed religious contexts. Again, this is a modeling choice. We shall further explore this issue using the university schema (Figure 2).

On one side, the whole schema deals with the academic context. If we insist on having this one context only, then every subclass relationship should be *category of* and, of course, the hierarchy would not have a forest structure. However, when the university environment is our application domain, having one university context does not help in comprehending the application. Naturally, we want to divide the schema into several contexts.

One of the contexts we identified in the university environment is the employment context. The major class for this context is **employee**. However, this is still quite a large context. If we are not dividing the employment context further then its hierarchy does not form a tree, since classes such as **teaching assistant**, **department-chairperson** and **academic administrator** have two superclasses each. For the double purpose of refined comprehension and to guarantee the forest structure of the schema we divide the employment context into two contexts, adding a teaching context, with the class **instructor** as a major class, to the general context of employment. Using these two contexts for employment causes the implied *category of* schema of Figure 12 to have the desired forest structure.

3.5 Applying the rules of disciplined modeling

Let us now reconsider Figure 2. We identify in the schema seven different hierarchical contexts. For each context we shall identify the major class of the context as mentioned in **RULE 3**.

1. Personal context: **person**. No descendants.
2. Learning context: **student**. Other classes: **grad-student, undergradstudent**.
3. Employment: **employee**. Other classes: **assistant, teaching assistant, research assistant, academic administrator, president, provost, college dean, department chairperson**.
4. Teaching: **instructor**. Other classes: **faculty member, special lecturer, professor, PhD advisor, adjunct**.
5. Alumni: **alumnus**. No descendants.
6. Publications: **publication**. Other classes: **refereed conference paper**.
7. Formal education: **formal education**. Other classes: **PhD degree, Bachelor degree**.

The partition of the schema into the above contexts implies the mapping of the subclass relationships into *category of* and *role of* relationships as shown in Figure 12.

In this process of dividing the employment context into two contexts we had to make some delicate decisions concerning the classes **teaching assistant**, **department-chairperson**, and **academic administrator**. E.g., for **teaching assistant** we had to decide whether it belongs to the employment context as its superclass **assistant** or to the teaching context as its superclass **instructor**. The dilemma can be resolved by **CASE 1.1** as teaching is the functionality of **teaching assistant** and thus **instructor** is the secondary superclass while **assistant** is the primary superclass as it is definitional. In fact, we can also apply **CASE 1.2** here as assistant is the noun and teaching is the adjective. Hence, a **teaching assistant** is an **assistant** with a *category of* relationship and thus belongs to the employment context, playing a *role of* an **instructor**.

The second dilemma is whether a **department chairperson** is in the teaching context or the academic administrator context. The main function of a chairperson is to manage the department. He also functions as a professor, e.g. in teaching a course, but this is a secondary function for him. Hence, by **CASE 1**, the class **department-chairperson** belongs to the academic administrator context.

The class **academic administrator** has superclasses **employee** and **professor**. The purpose of having a professor's appointment for an academic administrator is to provide an escape appointment for the case of resignation from the administrator position. Hence, by **CASE 1**, the primary (*category of*) superclass is

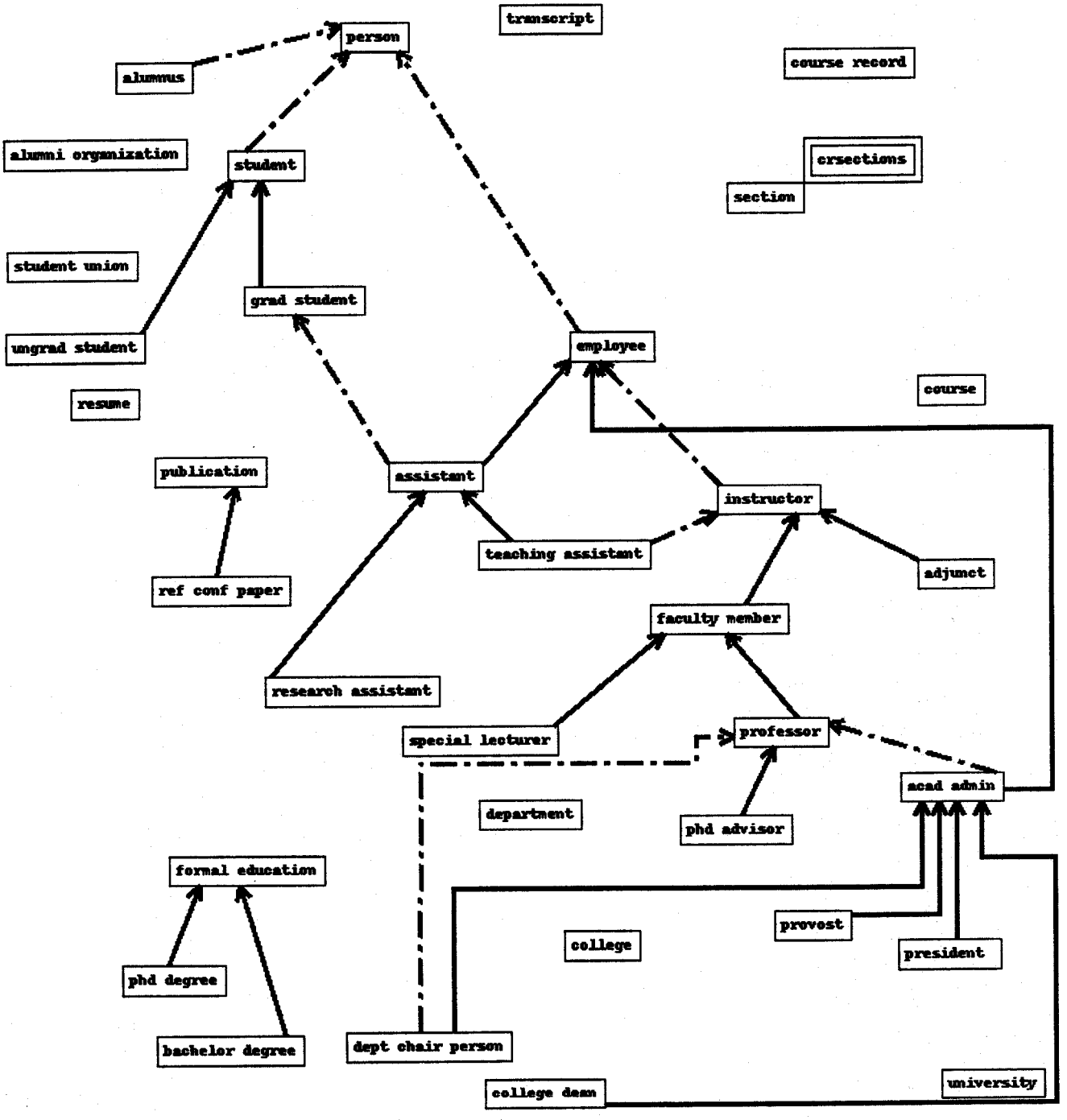

Figure 12. A subschema of a university database with category of and role of at display level 3)

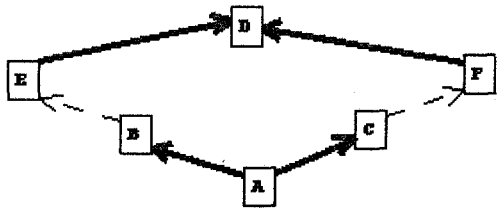

Figure 13. Schema demonstrating contradiction

employee and the secondary (*role of*) superclass is **professor**.

4 Category of hierarchy forms a forest aiding comprehension

4.1 Disciplined modeling results in a forest structure

The main purpose of this section is to prove the following theorem.

Theorem: Using *disciplined modeling*, a class has at most one *category of* superclass.

Such a theorem implies that the *category of* hierarchy has a forest structure, i.e., consists of one or several tree structures. These tree structures serve as the backbones of the schema and will be critical in the efforts to comprehend the schema and partition it into manageable subschemas.

PROOF: Suppose to the contrary that there exists a class A which is *category of* both class B and class C. Hence, A and B are in the same context. Similarly, A and C are in the same context. By the transitivity of the equicontext relation (**RULE 1**) B and C are in the same context.

By **RULE 3** the joint context of classes B and C has a root class D such that both B and C are *category of* descendents of D (Figure 13). In other words, there is a sequence of *category of* relationships from B (C) up to D. Let E (F) be a child class of the class D on the path of *category of* relationships from B (C) to D.

In the following discussion we are going to use a relation "*represents the same real world object*" defined for instances of a database. A pair of instances of the database belongs to this relation iff both instances represent the same real world object. This relation is obviously transitive.

Let a be an instance of class A. Then B has an instance b_a corresponding to instance a of A, since A is *category of* B. In other words, both instances a and b_a represent the same real world object. Similarly, there exists an instance c_a of C which represents the same real world object as the instance a of A. Thus both instances b_a and c_a represent the same real world object, due to the transitivity of "*represents the same real world object.*"

As noted before, B (C) has a sequence of *category of* relationships up to E (F). Hence, E (F) contains an instance e_a (f_a) corresponding to the instance b_a (c_a) of class B (C) where this correspondence is defined transitively from the correspondence along the sequence of *category of* relationships. Hence, both e_a and b_a (c_a and f_a) represent the same real world object. But, b_a and c_a represent the same real world object. Thus it follows from the transitivity of "*represents the same real world object*" that e_a and f_a represent the same real world object. But by **RULE 2** the extents of the classes E and F which are both *category of* D may not both contain an instance representing the same real world object, so the previous conclusion contradicts our assumptions. Hence, A cannot be *category of* B and C, but A can be only *category of* one class. ■

4.2 Utilizing forest structure for schema understanding

Figure 14 shows the forest hierarchy of the *category of* relation of Figure 12. This schema shows the different contexts as trees in the forest. Thus, it gives a concise initial representation of Figure 1. Provided with this comprehension the user is ready to study the intricacies of the schema, by first comprehending the *role of* relationships connecting different contexts as shown in Figure 12. Then he should be ready to cope with the relationships. To this aim, the user picks one context at a time and concentrates on the relationships between classes of this context and classes of the other contexts, one at a time.

Consider, e.g., the interactions between the teaching context, and other contexts. We find interactions with the contexts employment, learning and some singleton classes. We refer to relationships pointing from one context to another as *inter-relationships*. The inter-relationships with the employment context are: **instructor** is *role of* **employee** and **teaching assistant** is *role of* **instructor**, and **professor** is Supervisor of **assistant**.

We can now divide the relationships of a schema into subsets as follows. For every context there is a set of relationships that are completely contained in the context, which we call *intra-relationships*. The inter-relationships can be divided into groups according to

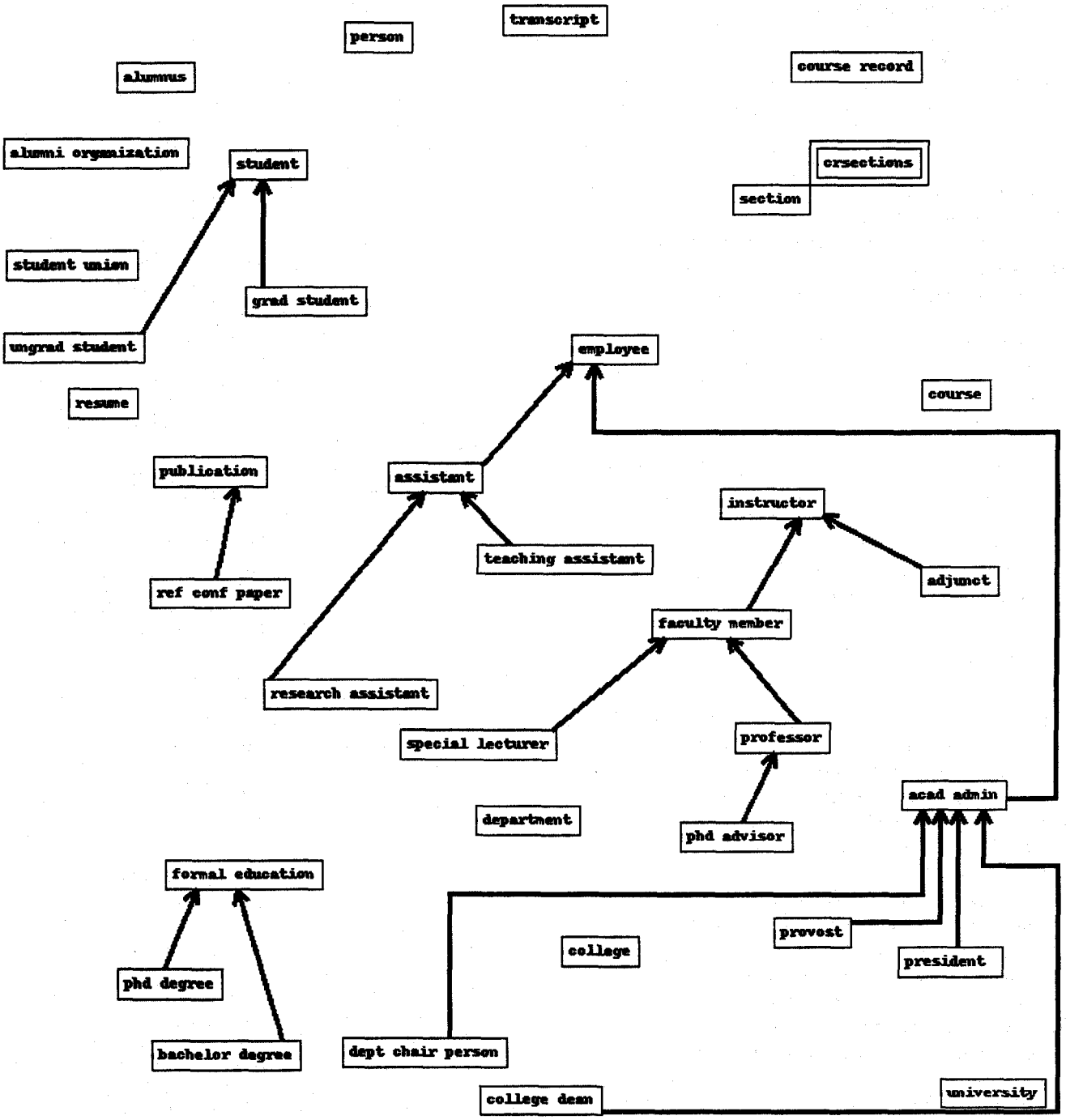

Figure 14. The forest induced by the university database

which other context or singleton class they are coming from or going to. In this way, we divide the task of studying all relationships into a number of smaller, well organized tasks. This process will help dramatically the comprehension process of the user, due to the reduction in the complexities of the subschemas versus the complexity of the original schema as demonstrated now.

Consider a schema containing ten contexts, with n classes in each context. Suppose further that the number of relationships within each context is αn and the number of relationships connecting each pair of contexts is βn. Then the complexity of the schema is

$$\mathbf{c} = (10\alpha n + (10 * 9/2)\beta n)/10n = \alpha + 4.5\beta$$

On the other hand when considering only a subschema of two contexts, the complexity is

$$\mathbf{c} = (2\alpha n + \beta n)/2n = \alpha + \beta/2$$

Hence, the complexity of considering each time only the inter-relationships between two contexts is reduced by a factor of one less than the number of contexts in the schema. Hence, each comprehension task is not only small in magnitude but also much less complex than the task of comprehending the schema as a whole.

5 Summary

In this paper we present a theoretical paradigm to support the existence of a meaningful forest subhierarchy of a given specialization hierarchy. Such a forest hierarchy functions as a skeleton of the original schema and supports comprehension efforts. The extraction of the forest subschema employs two approachs which we apply in this research: *informational thinning* and *partitioning*. We introduce a new technique for modeling called *disciplined modeling*. It is based on three rules which expresses limitations and refinements to the modeling of the schema. These rules are needed to prove the existence of a forest subschema whose trees represent logical units. Our techniques are demonstrated on a university environment schema. In a subsequent paper we will present a methodology for finding such a forest subschema, based on the theoretical paradigm. This methodology will be applied to a medical vocabulary subschema of the MED.

6 Acknowledgement

We thank J. Cimino, M. Halper and E. Neuhod for their important feedback on earlier drafts of this paper.

References

[1] H. Gu, Y. Perl, J. Geller, J. Cimino, M. Halper, and M. Singh. A methodology for finding a central forest hierarchy in an OODB specialization hierarchy, 1996. In preparation.

[2] J. Geller, Y. Perl, P. Cannata, A. Sheth, and E. Neuhold. Structural integration: Concepts and case study. *Journal of System Integration*, 3(2):131–161, 1993.

[3] J. Geller, Y. Perl, E. J. Neuhold, and A. Sheth. Structural schema integration with full and partial correspondence using the dual model. *Information Systems*, 17(6):443–464, 1992.

[4] J. Cimino and G. Barneet. Automated translation between medical terminologies using semantic definitions. *MD Comput.*, 7:104–109, 1990.

[5] J. Cimino, P. Clayton, G. Hripcsak, and S. Johnson. Knowledge-based approachs to the maintenance of a large controlled medical terminology. *Journal of the American Medical Informatics Association*, 1(1):35–50, 1994.

[6] M. Halper, J. Geller, and Y. Perl. Part relationships for object-oriented databases. In G. Pernul and A. Tjoa, editors, *Proceedings of the 11th International Conference on the Entity Relationship Approach*, pages 406–422, Karlsruhe, Germany, 1992.

[7] M. Halper, J. Geller, and Y. Perl. An OODB part relationship model. In Yelena Yesha, editor, *Proceedings of the First International CIKM Conference*, pages 602–611, Baltimore, MD, 1992.

[8] Ashish Mehta, J. Geller, Y. Perl, and P. Fankhauser. Computing access relevance to support path-method generation in interoperable Multi-OODB. In *RIDE-93 (Research Issues in Data Engineering)*, pages 144–151, Vienna, Austria, 1993.

[9] A. Mehta, J. Geller, Y. Perl, and P. Fankhauser. Computing access relevance for path-method generation in OODBs and IM-OODB. *Journal of Inteligent Information System*, 1996.

[10] Ashish Mehta, James Geller, Yehoshua Perl, and Erich Neuhold. The OODB Path–Method Generator (PMG) using precomputed access relevance. In *Proceedings of the 2nd Int'l CIKM Conference*, pages 596–605, Washington D.C., 1993.

[11] Peter Pin-Shan Chen. The Entity-Relationship Model: Toward a unified view of data. *TODS*, 1(1):9–36, 1976.

[12] J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy, and W. Lorensen. *Object-Oriented Modeling and Design*. Prentice-Hall, Inc., Englewood Cliffs, NJ 07632, 1991.

[13] Grady Booch. *Object-Oriented Analysis and Design with application*. The Benjamin/Cummings Publishing Company, Inc., Redwood City, CA, 1994.

[14] M. Halper, J. Geller, Y. Perl, and E. J. Neuhold. A graphical schema representation for object-oriented database. In R. Cooper, editor, *Workshop on Interfaces in Database Systems (IDS-92)*, pages 282–307. Springer Verlag, London, 1992.

[15] M.R. Gary and D.S. Johnson. *Computers and Intractability*. Freeman, New York, 1979.

[16] E. Agasi, R.I. Becker, and Y. Perl. A shifting algorithm for constrained min-max partition on trees. *Discrete Applied Mathematics*, 45:1–28, 1993.
[17] R.I. Becker and Y. Perl. The shifting algorithm technique for the partitioning of trees. *Discrete Applied Mathematics*, 62:15–34, 1995.
[18] R.I. Becker, Y. Perl, and S. Schach. A shifting algorithm for min-max tree-partitioning. *J. ACM*, 29:56–67, 1982.
[19] S. Kundu and J. Misra. A linear tree-partitioning algorithm. *SIAM J. Comput.*, 6:131–134, 1977.
[20] Y. Perl and S. Schach. Max-min tree-partitioning. *J. ACM*, 28:5–15, 1981.
[21] J. Geller, Y. Perl, and E. Neuhold. Structure and semantics in OODB class specifications. *SIGMOD Record Special issue on Semantic Issues in Multidatabase Systems*, 20(4):40–43, 1991.
[22] E. J. Neuhold and M. Schrefl. Dynamic derivation of personalized views. In *Proceedings of the 14th International Conference on Very Large Databases*, pages 183–194, Long Beach, CA, 1988.
[23] Saša Buvač and Richard Fikes. A declarative formalization of knowledge translation. In *Fourth International Conference on Information and Knowledge Management (CIKM)*, pages 340–347, Baltimore, MD, 1995.
[24] Saša Buvač and Ian M. Mason. Propositional logic of context. In *Proceedings of the 11th National Conference on Artificial Intelligence (AAAI-93)*, pages 412–419, Washington, DC, 1993.
[25] R. V. Guha. Contexts: A formalization and some applications, 1991.
[26] Lucja Iwanska. Context in natural language processing. In *Working Notes of Workshop W13, IJCAI*. Montreal, Canada, 1995.
[27] John McCarthy. Notes on formalizing context. In *13th International Joint Conference on Artificial Intelligence*, pages 555–560, Chambery, France, 1993.
[28] George A. Miller. Wordnet: A lexical database for english. *Communications of ACM*, 38(11):39–41, 1995.
[29] Y. Shoham. *Varieties of Context Artificial Intelligence and Mathematical Theories of Computation*. Academic Press, London, 1991.
[30] D.B. Lenat. CYC: A large-scale investment in knowledge infrastructure. *CACM*, 38(11):33–38, 1995.
[31] L. Shastri. *Semantic Networks: an Evidential Formalization and its Connectionist Realization*. Morgan Kaufmann Publishers, San Mateo, CA, 1988.
[32] L. Shastri. Default reasoning in semantic networks: a formalization of recognition and inheritance. *Artificial Intelligence*, 39(3):283–356, 1989.