ABSTRACT

MODELING CONTROLLED VOCABULARIES USING OODBS AND MULTILEVEL AREA DIAGRAMS

by
Li-min Liu

A Controlled Vocabulary (CV) is a software system of domain knowledge that consolidates and unifies the terminology of a large application domain. With a common, centralized CV, costly and time-consuming translations can be eliminated between pairs of organizations and pairs of software systems. Unfortunately, the more knowledge we put into a CV, the harder it is to understand and maintain it. In this dissertation, a comprehensive theoretical methodology for modeling CVs using Object-Oriented Database (OODB) technology is presented. We present two methods for representing a semantic network CV as an equivalent OODB, which we call an Object-Oriented Vocabulary Repository (OOVR). The first method, based on a structural analysis and partitioning of the CV, yields an OODB with a very concise schema, referred to as the OOVR schema. Due to its compact size, the schema can be displayed on one or a few computer screens and serves as an aid for comprehending and maintaining the CV. A program called the Object-Oriented Vocabulary Repository Generator (OOVR Generator) has been built to automatically generate an OOVR for a given semantic network CV. Our second methodology results in a larger schema, which, however, serves as an important tool for browsing and navigation through a CV. The OODB schemas created by both methodologies provide important abstract views of CVs. We have also defined a new type of semantic relationships called IS-A$'$ in the context of an OOVR representation. The IS-A$'$ relationships are defined on OOVR schemas to reflect certain important IS-A relationships in the underlying CV. The two OOVR representations exhibit several interesting theoretical characteristics which are formally proven in this dissertation.

To provide an environment with several abstract views of a CV, we also define a paradigm called Multilevel Area Diagrams (MLADs). A MLAD is a collection of different partitions of increasing detail and decreasing abstraction derived from a CV. Users can browse at one level and then switch to another level to continue their navigation. Examples of browsing sessions are presented to show that the MLAD paradigm provides processing capabilities beyond those of a triditional object-oriented representation of a vocabulary.

MODELING CONTROLLED VOCABULARIES USING OODBS AND MULTILEVEL AREA DIAGRAMS

by
Li-min Liu

A Dissertation
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy

Department of Computer and Information Science

May 1999

APPROVAL PAGE

Modeling Controlled Vocabularies using OODBs and Multilevel Area Diagrams

Li-min Liu

Dr. Yehoshua Perl, Dissertation Advisor — Date
Professor of Computer and Information Science,
NJIT, Newark, NJ

Dr. James Geller, Co-Advisor and Committee Member — Date
Associate Professor of Computer and Information Science,
NJIT, Newark, NJ

Dr. James J. Cimino, Committee Member — Date
Associate Professor of Medical Informatics,
Columbia University, New York, NJ

Dr. Michael Halper, Committee Member — Date
Assistant Professor of Mathematics and Computer Science,
Kean University, Union, NJ

Dr. Waldemar G. Johanson, Committee Member — Date
Chairman and Professor of Medicine,
New Jersey Medical School, UMDNJ, Newark, NJ

BIOGRAPHICAL SKETCH

Author:	Li-min Liu
Degree:	Doctor of Philosophy
Date:	May 1999

Undergraduate and Graduate Education:

- Doctor of Philosophy in Computer Science,
  New Jersey Institute of Technology, Newark, NJ, 1999
- Master of Science in Computer and Information Science,
  Syracuse University, Syracuse, NY, 1995
- Bachelor of Science in Computer Science,
  Tamkang University, Taipei, Taiwan ROC, 1990

Major: Computer Science

Presentations and Publications:

L. Liu, M. Halper, J. Geller and Y. Perl, Controlled Vocabularies in OODBs: Modeling Issues and Implementation, *Distributed and Parallel Databases*, 7(1):37-65, January, 1999.

L. Liu and M. Halper, Incorporating Semantic Relationships into an Object-Oriented Database System, in *the Proceeding of Thirty-Second Hawaii International Conference on System Sciences (HICSS-32)*, January, 1999.

H. Gu, L. Liu, M. Halper, J. Geller and Y. Perl, Converting an Integrated Hospital Formulary into an Object-Oriented Database Representation, in *the Proceedings of the 1998 American Medical Informatics Association Annual Fall Symposium (AMIA'98)*, pages 770-774, Orlando, FL, November 1998.

L. Liu, M. Halper, H. Gu, J. Geller and Y. Perl, Modeling a Vocabulary in an Object-Oriented Database, in *Proceedings of Fifth International Conference on Information and Knowledge Management (CIKM-96)*, pages 179–188, Rockville, MD, November 1996.

This work is dedicated to my family

謹以此博士論文獻給我最敬愛的父母親及家人

ACKNOWLEDGMENT

First of all, I would like to extend my sincere gratitude to my advisor Dr. Yehoshua Perl for his guidance. It would have been impossible to do this dissertation without him. Next, let me extend my thanks to Dr. James Geller and Dr. Michael Halper. They constantly gave me support, encouragement, and reassurance. Without the help from all three of them, I simply would not have completed this work. Next, I thank Dr. James Cimino and Dr. Waldemar Johanson for serving as members of the committee.

I would like to thank Dr. Ching-Yu Huang and Dr. Gung-Wei Chirn for giving me counsel and helping me settle down at the beginning of my studies at NJIT. Thanks to my friends at NJIT: Dr. Shy-Shyan Chen, Dr. Yui-Lian Chen, Dr. Chih-Ying Wang, Jui-Yuan Ku, George Chang, Pao-Lien Wang, and Feng-Shen Kuo. You all made my life more cheerful at NJIT.

Thanks to my fellow doctoral student Huanying Gu. It was my pleasure to work with you on the OOVR project. I also thank the members of the AI and OODB Lab: Dr. Yugyung Lee, Wenguang He, Zong Chen, and Roberto Galnares as well as all Masters students who graduated from the lab. Thank you for all my joyful days in the lab; it was my privilege to work with you all.

Finally, I would like to thank my family, to whom this dissertation is dedicated: my father Yi-Tang, my mother Chi-May Tasi, my brother Lo-min, and my sister-in-law Shu-Hua Lai. Thanks for your never-ending support, understanding, and encouragement. Also thanks to my two nephews, 5-year-old Martin and 3-year-old Matthew, who remind me that life can be very simple and full of happiness.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Figure Page

Figure Page

CHAPTER 1

INTRODUCTION

A Controlled Vocabulary (CV) is a software system of domain knowledge that consolidates and unifies the terminology of a large application domain [13, 68]. With a common, centralized CV, costly and time-consuming translations can be eliminated between pairs of organizations and pairs of software systems. For example, in the healthcare field cases have been reported of differences between terminologies used by different laboratories within the same hospital [11]. Any industry with such a problem needs a CV to overcome the communication difficulties between multiple cooperating organizations. CVs can provide solutions for the following two aspects of this problem. From an application standpoint, controlled vocabularies alleviate software systems of the burden of maintaining their own *ad hoc* vocabularies. From a user point of view, controlled vocabularies help in standardizing information processing among different organizations, thus reducing the overall cost of doing business. Unfortunately, it is difficult to understand and maintain a CV when it contains an enormous amount of knowledge.

In this dissertation, we develop theoretical techniques and software tools to model a large CV in order to help users comprehend the contents of the CV, as well as make it more easily maintainable. In order to do so, we utilize Object-Oriented Database [3, 44, 54, 70, 111] (OODB) modeling and technology. Specifically, we present a method for representing a CV as an equivalent OODB (which we call an OOVR.) A major advantage of this approach is the fact that the OODB schema provides an important abstract view of the CV. We will, in fact define a framework which provides many levels of abstraction.

The basis of our methodology is the notion of "area," which can be used to partition a large CV into manageable units. In this context, we have defined and utilized the notions of property-introducing concept and intersection concepts. The partitioning of the CV eventually leads to the definition of an OODB schema, which we call the OOVR schema. The schema is an important abstract view of the distribution of properties and the inheritance that takes place within the CV.

In [39, 41], the authors described how to use the OOVR schema to help the CV designer correct errors in the CV. The schema also helps a user comprehend the of a large, complex vocabulary [39, 41] because the number of classes in the schema is much smaller than the number of concepts in the underlying vocabulary network. Due to its compact size, the schema can be displayed on one or a few computer screens. On the other hand, it is almost impossible to display a typical CV in such a space, because it may contain thousands of concepts. We will also demonstrate how the schematic representation of the CV aids in browsing and traversal.

We have developed a software system called the *OOVR Generator* which automatically carries out all phases of mapping a CV to a OOVR representation. It takes as its input a CV in a "flat file" semantic network format and produces an equivalent, fully populated OOVR as its output. Both the architecture of the OOVR Generator and the functionality of every component will be described.

In the course of using the OOVR schema, we realized that certain refinement were needed with respect to a type of area called a multi-rooted intersection area. Specifically, we devised an additional methodology for further partitioning such areas. The methodology has the recursively defined notion of articulation concept as its basis. From this, we define partitioning units called partial areas. The result of our revised methodology is the singly-rooted schema.

After we examined a special type of intersection area, the "multi-rooted intersection area," we discovered some difficulties in constructing SUBCLASS_OF relationships for these area classes. We also discovered some difficulties in browsing multi-rooted intersection area classes in an OOVR schema. In order to overcome these difficulties, we present a refined modeling technique. We first identify *articulation concepts* in a multi-rooted intersection area and then use them to generate *partial areas*. An articulation concept serves as a naming concept to name the partial area and define the SUBCLASS_OF relationships of the partial area class. The result of this is a singly-rooted OOVR representation. The schema of this representation is called the singly-rooted schema.

The singly-rooted schema provides not only a structural view but also an enhanced semantic view of a CV. This is because partial areas (derived from one multi-rooted intersection area A) have the same set of properties (structure), and all concepts in an area (and in a partial area) are descendents of a single concept (a root), which makes them semantically similar. Compared to the original OOVR schema, the singly-rooted schema is a more detailed abstraction network. Both the original OOVR representation and the singly-rooted representation exhibit some interesting theoretical characteristics. We will be stating a number of these as theorems and lemmas, along with proofs.

We have applied our OODB modeling techniques to several existing vocabularies, including the InterMED and the MED. The results of these mappings will be discussed. Both of these representation called the InterMED OOVR and MED OOVR, respectively, are up and running on top of ONTOS, a commercial OODB system. The InterMED OOVR is accessible via the Web [85].

Another important contribution of our research is the introduction of a new kind of semantic relationship, called an IS-A$'$ relationship, into the OOVR representation. The IS-A$'$ relationships are defined on OOVR schemas to reflect certain important IS-A relationships in the underlying CV. Users can navigate an OOVR schema through not only the SUBCLASS_OF relationships but also our new IS-A$'$ relationships. With IS-A$'$ relationships, users are offered an enhanced abstraction view of a CV in order to improve navigation and general usage.

Both kinds of OOVR schemas, the original OOVR schema and the singly-rooted schema, provide valuable views of the knowledge content of a CV. Due to this, users may want to have both abstractions available during a browing session of an OOVR. To achieve this goal, we introduce a paradigm called Multilevel Area Diagrams (MLADs). An MLAD is a collection of different partitions (area diagrams) derived from a CV and relationships among these. Users can browse in one area diagram first and then switch to another area diagram to continue their navigation. Users can also switch from an area diagram directly to the underlying CV at any time. Examples of browsing sessions are described to show that the MLAD paradigm provides capabilities beyond those of the basic OOVR representation.

There are a number of reasons why one would want to model a CV in an OODB form, in addition to those mentioned above. First, most OODB systems support object-oriented programming languages such as Smalltalk [36] and C++ [100, 101]. When applications are developed in object-oriented programming languages, there is a low "impedance mismatch" path to an OODB [110]. Once a CV is modeled in an OODB representation, the vocabulary can also be accessed declaratively using an SQL extension (like OSQL of ONTOS [84]) or a "path" language such as XQL [53]. If one would like to access the vocabulary remotely, the Common Object Request Broker Architecture (CORBA) [76] can be used for this purpose.

Additionally, from a theoretical standpoint, the typical OODB system's repertoire of modeling constructs neatly captures many modeling features of semantic networks which typically are used to describe a vocabulary. Thus, the vocabulary can be mapped directly from the semantic network into the OODB system without having to re-model it from scratch.

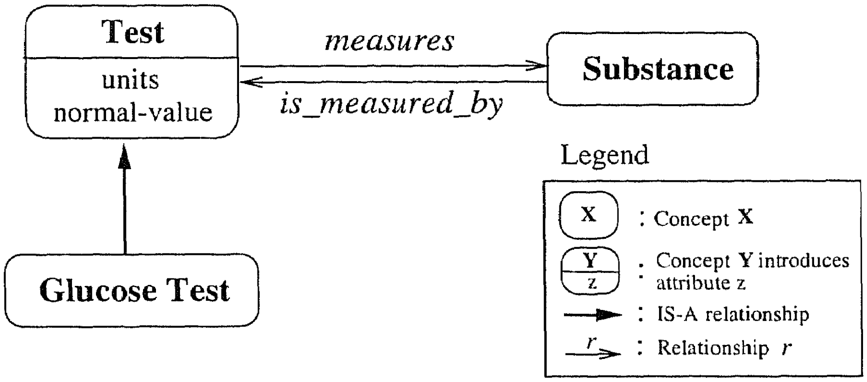


Figure 1.1 Small extract of a CV

1.1 The Structure of a CV

In this dissertation, a CV is a semantic network of concepts (nodes), with attributes, connected by IS-A links and/or semantic relationships. An attribute is a property whose value is a primitive data type (such as a string). One attribute common to all nodes is *name*, which holds a concept's associated *term* (or textual denotation) [27]. Another common attribute is *synonyms* which can hold alternate denotations aside from the primary one. A relationship has as its value a reference to another concept in the network. IS-A links denote the specialization relationship between concepts. IS-A links also enable the property inheritance mechanism within the network. In this dissertation, a CV can be understood and drawn as a directed acyclic graph (DAG). We will be using the following graphical conventions when drawing the elements of a CV. A concept is a rectangle having rounded corners with its name (term) written inside. The names of any attributes introduced by the concept (when shown) are written below the concept's name and are separated from it by a line. Note that the values of such attributes will not be included in any diagrams. A relationship is a labeled arrow directed from the source concept to the target concept.

Figure 1.1 shows a small extract of a CV with three concepts: **Test**,[1] **Glucose Test**, and **Substance**. The concept **Test** introduces the two attributes *units* and *normal-value* and the relationship *measures* directed to **Substance**. The concept **Substance** introduces the relationship *is-measured-by* (the converse of *measures*) but no attributes. The **Glucose Test** is a subconcept of (or, simply, "IS-A") **Test**, and therefore it inherits all of **Test**'s properties.

1.2 Related Work

Computerizing natural language concepts has long been a major goal of computer scientists. Various forms of semantic networks [4, 63, 95, 96, 108], knowledge representation languages [52, 73, 78], ontologies [83], and semantic data models [45, 47] have been recruited to tackle this task. One kind of semantic network called by its creator "conceptual graph" is described in [93, 94]. Conceptual graphs contain additional components, besides the semantic network. There exist several general ontologies such as CYC [17, 65, 66] and WordNet [74, 109]. CYC is a general ontology for common sense knowledge to facilitate reasoning. It contains more than 10,000 concept types and is rooted in the concept "Thing" which does not have any properties. WordNet is an on-line lexical reference system. It is a taxonomy which has no structured concepts or axioms. English nouns, verbs, adjectives and adverbs are organized into sets of synonyms (synsets). Each synset represents a lexicalized concept. WordNet includes six kinds of semantic relations among synset. They are Synonymy, Antonymy, Hyponymy, Metonymy, Toponymy, and Entailment. For example, "pipe" and "tube" are linked by the Synonymy relation. WordNet presently contains over 90,000 synsets and 116,000 occurrences of semantic relations.

[1] Some typographical conventions: A bold face font will be used for concepts' terms. Properties of concepts will appear in italics and will be written strictly in lowercase letters. Object classes will start with uppercase letters.

Aside from general ontologies, many domain specific ontologies exist such as those for the TOVE enterprise model [28, 29, 30, 38, 103]. The TOVE (TOronto Virtual Enterprise) project focuses on supporting enterprise integration. Rather than having a single ontology, there are several ontologies for various logical parts of the enterprise model. In the TOVE ontology, the basic entities are represented as objects with specific properties and relations. Objects are structured into taxonomies. The definitions of objects, attributes, and relations are specified in first-order logic. The methodology for building, as well as the framework for evaluating, an ontology for enterprise modeling can be found in [38].

The medical field has seen the introduction of a number of CVs. These include SNOMED [14], SNOMED II [14, 56], SNOMED RT [56, 97], ICD9-CM [106], MED [12, 13], the GALEN project's Core Model [89] (as expressed in GRAIL [35]), MeSH [80], CPM93 [15], and CPT98 [1] many of which have been integrated into the UMLS [48, 67, 104, 105].

The SNOMED is the Systematized Nomenclature of Human and Veterinary Medicine. "It is a detailed and specific coded vocabulary of names and descriptions used in healthcare" [56]. Terms in SNOMED are arranged in a hierarchy. Each term has been assigned to a term code where each digit represents a specific location in the hierarchy. SNOMED RT (Reference Terminology) is designed to complement the broad coverage of medical concepts in SNOMED. A reference terminology "is a set of concepts and relationships that provides a common reference point for comparison and aggregation of data about the entire healthcare process, recorded by multiple different individuals, system, of institutions" [97]. In SNOMED RT, the code for each term no longer carries the hierarchical meaning. A series of relational tables with explicit relationships between terms and their parent terms is provided. A joint project between Kaiser Permanente and the Mayo Clinic to develop a Convergent Medical Terminology (CMT) is based on SNOMED [8].

ICD9-CM stands for "The International Classification of Diseases, 9th Revision, Clinical Modification." The ICD9-CM is maintained jointly by the National Center for Health Statistics and the Health Care Financing Agency. It is designed for the classification of information for statistical purposes, and for the indexing of hospital records by disease and operations. It is also used for data storage and retrieval. ICD9-CM determines the Diagnosis Related Group (DRG) code which indicates what was wrong with and what was done for a patient. A DRG controls reimbursement by U.S. Public Health Service and Health Care Financing Administration programs. ICD9-CM is organized into two classification trees: one for diagnoses and the other for procedures. The diagnoses classification tree contains 4 levels of depth, and the procedures tree has 5 levels of depth. No relationships or attributes have been defined in it.

National Library of Medicine's Unified Medical Language System (UMLS) project is focusing on helping health professionals and researchers retrieve and integrate electronic biomedical information from a variety of sources, irrespective of the variations in the way similar concepts are expressed in different sources. It contains four components called knowledge sources: Metathesaurus, Semantic Network, SPECIALIST Lexicon, and Information Sources Map. The Metathesaurus is organized by concepts, terms, and strings. Alternate names for the same concept (synonyms, lexical variants, and translations) are linked together. The 1998 version of the Metathesaurus contains 476,322 biomedical concepts with 1,051,903 different concept names from more than 40 source vocabularies. Concepts are also linked by relationships, some of which are derived from the source vocabularies; others are created during the construction of the Metathesaurus. The Semantic Network specifies the types of concepts in the Metathesaurus. Each concept is assigned to at least one type in the Semantic Network. To some degree, the Semantic Network is similar to an OODB schema if we consider concepts in the Metathesaurus as instances of an OODB. However, a concept may have more than one specified type; the UMLS maintains extra mapping entries for those kinds of concepts rather than creating multiple inherited types. In the OODB paradigm, an instance is prohibited from belonging to more than one class.

The GALEN (Generalized Architecture for Languages Encyclopaedias and Nomenclature in Medicine) project [89] aims to create a multilingual "terminology server." GRAIL (GALEN Representation and Integration Language) [35] is a descriptive logic with subsumption and multiple inheritance based on semantic networks. It belongs to the KL-ONE family [5]. The GRAIL is used in a prototype clinical workstation, PEN&PAD [33]. One may use GRAIL to build a terminology in domains other than the medical field [33]. GRAIL has been used to develop the prototype GALEN COmmon REference (CORE) model for medical terminology [34]. Unlike other systems, for building medical terminologies, GRAIL has "category" and "individual" layers which are roughly equivalent to "class" and "instance" layers in OO modeling [88]. However, the ability to generate more layers at different levels of abstraction is missing in this language.

An object-oriented framework has previously been employed as a modeling platform for thesauri used in (natural) language-to-language translation [25, 26]. TEDI, a terminology editor, was built in the same context as a tool for extracting relevant information from hypermedia documents [75]. The O_2 OODB system [18, 19, 92, 102] has been used to store portions of a general English dictionary based on a "feature structure" description of its entries [49]. In a similar effort, [112] presents a technique for storing a dictionary in an ObjectStore database [62, 92]. Srinivasan proposed a general approach and program to build a controlled vocabulary in the C programming language [98]. However, the information is not stored in an object-oriented framework. The user has no way of viewing the information at an abstract level.

The abstraction techniques in semantic modeling [86, 87] consider the higher level nodes in a hierarchy to be the higher level abstractions. A higher/lower level abstraction is generated by climbing up/down along a particular semantic relationship such as IS-A or part-of. Therefore, there exists no overall abstraction for the whole hierarchy in semantic modeling.

Object Lens [61] is a system that integrates facilities for hypertext [16], object-oriented databases, electronic messaging and rule-based agents. It is a successor to the Information Lens [71, 72]. Object Lens can create, modify, retrieve, and display objects that represent physical or conceptual entities such as messages, people, meeting, etc. For displaying the contents of an extension of a certain object type, Object Lens provides two formats: *tables* and *trees*. For each object type, Object Lens provides a form (template) to help users in editing objects. Associations between objects are represented by links which are hyperlinks in a form.

A descriptive semantic network called Structured Meta Knowledge (SMK), employing a terminological knowledge-base, has been used to capture the semantics of patients' medical records [35]. SMK, which arose from the PEN&PAD project [90], has three levels of abstraction: category, individual, and occurrence. All occurrences are associated with a *time*, *place*, and *agent*. For example, "Patient," "Trauma," and "Bone" are entities on the category level. "Jane Smith" and "Fracture which" are entities on the individual level. "Jane Smith asSeenBy Dr. Peters at City HC on 4th July 1990" is an entity on the occurrence level [35].

The Terminological Knowledge Representation System (TKRS) consists of a T (terminology) Box (TBox), where concepts are introduced, and an A (assertion) Box (ABox) or world description , where facts about individuals are stated in terms of concept memberships [7]. The major two functions of TBox are (1) to declare frame-like structures by introducing primitive concepts and roles, and (2) to define new concepts in terms of primitive ones by specifying necessary and sufficient conditions for concept membership [6, 7]. These two functions in the TBox are separated into two parts called the schema and the view. In other words, a TKRS has been refined into three parts: the schema, the view taxonomy, and the world description. Note that the notation of schema here is not the same as for OODBs. Also, the notation of view is different from the view system [46, 57, 58, 59, 60] in OODB systems.

CDME (Collaborative Device Modeling Environment) is a Web-based compositional modeling system [22] aimed at sharing and collaborative construction of knowledge bases [50]. CDME has been implemented at three levels: *physical*, *ontological*, and *logical*. Each level has its own representation language, namely, CML [21], Ontolingua [37], and KIF [32], respectively. At the ontological level of CDME, an ontology defines the vocabulary in some domain of discourse. Support of human understanding and interaction is one use for ontologies [23]. "However, as a CML domain theory gets larger, it becomes more difficult to understand its context and implicit assumptions, and harder to reuse" [50]. The Ontolingua Server has been built on an extended version of Ontolingua [37]. The overall design goal for the Ontolingua Server was to facilitate the collaborative development of ontologies [23]. The Ontolingua Server is a central server which uses HTTP [24] to communicate with remote users logged on via a Web browser. It has more than 1,000 users of which 150 are described as serious [23].

Database technology has been used as a means for bringing persistence to knowledge-based systems. HYWIBAS [82] is a knowledge-based system which supports three levels of semantic constructs- frames, objects, and relations. This system adopts a two-level mapping: from a frame-based knowledge representation model to an object-oriented data model (COCOON [91]), and from COCOON to a relational system. INGRES [99] is used in HYWIBAS. "COCOON may be considered as lying somewhere between prescriptive and descriptive paradigms" [82]. The two basic representation structures in COCOON are type and class. A COCOON class represents a descriptive grouping of objects and may have an associated predicated condition. A COCOON type describes what properties an object can have. In this way, a COCOON type is similar to a class in an OODB system. However, there is no hierarchy among COCOON types. If we map COCOON types into an OODB, we will generate a flat schema (i.e., one having no relationships among classes).

In [51], the EXODUS object manager [9] is used as a subsystem of a frame representation system [52]. A storage model, based on techniques previously proposed for OODBs [107], has been employed as the basis for storing Telos knowledge-bases on disk [55, 77, 78, 79]. Both these efforts sought to incorporate their database subsystems transparently.

1.3 Outline of the Dissertation

The remainder of this dissertation is organized as follows. In Chapter 2, brief descriptions of the MED and InterMED (two existing large CVs) are presented and the mapping methodology which creates the OOVR from a CV is described. The architecture of the OOVR Generator and other implementation issues are also discussed. In Chapter 3, we first provide an alternative presentation of the development of the OOVR supporting it with a theoretical framework. Then we describe the difficulties caused by multi-rooted intersection areas in detail. This is followed by the revised mapping methodology. We also analyze and prove certain theoretical characteristics of both OOVR representations.

Chapter 4 discusses problems which may exist for the singly-rooted schema and how these problems affect the browsing of the schema. Then, we define the semantic relationship called "IS-A$'$" to overcome these problems. Examples with IS-A$'$ relationships are provided. The singly-rooted schema with IS-A$'$ relationships exhibits several interesting characteristics which are presented as lemmas and theorems with proofs.

In Chapter 5, we present the notion of Multilevel Area Diagram, which as an integrated system, provides users with more than one abstraction view of a CV. Use of MLADs is demonstrated.

In Chapter 6, we conclude with a summary and a discussion of future work. Preliminary and shorter versions of the work presented in this dissertation may be found in [42, 68, 69].

CHAPTER 2

REPRESENTING A CONTROLLED VOCABULARY AS AN OODB

In this chapter, we first introduce our two test-bed vocabularies, the MED and the InterMED. After that, we present some possible OODB modeling approaches for a CV. We then go on to describe our novel approach to represent a CV as an OODB. We develop the OOVR in two steps, without intersection concepts (areas) and with intersection concepts (areas). Finally, we describe the architecture of the OOVR Generator and its various components in detail.

2.1 Description of the InterMED and MED

InterMed Collaboratory is a collaborative project involving six participating medical institutions, Stanford University, Columbia University, Brigham and Women's Hospital, Massachusetts General Hospital, McGill University, and the University of Utah, with funding from the National Library of Medicine (NLM). Software and tools have been built for the merging, managing, and searching of a standardized vocabulary in a network-based entities dictionary. In this document, the vocabulary built under the InterMed project is referred to as the InterMED. The InterMED contains a small subset of the MED (Medical Entities Dictionary) which was developed and is presently in use at Columbia-Presbyterian Medical Center (CPMC) [12].

The MED and InterMED feature a concept subsumption hierarchy—a directed acyclic graph (DAG) composed of concepts connected through sub-concept (IS-A) and super-concept links. The IS-A hierarchy serves two main purposes. First, as we mentioned earlier, it supports the inheritance of properties among concepts within the CV. A subconcept is defined to inherit all the properties of its superconcept(s).

Because the IS-A hierarchy is a DAG, a concept may have more than one superconcept (or parent). In such a case, the subconcept inherits from all its parents. The second purpose of the hierarchy is to support reasoning in the form of subsumption-based inferences. For example, using the fact that **Tetracycline** IS-A **Antibiotic**, a decision-support system can infer that a patient is on antibiotics from an entry in a clinical database stating that the patient is taking tetracycline. Note that the IS-A hierarchy is singly-rooted. The roots of the MED and InterMED are **Medical Entity** and **Entity**, respectively.

The MED comprises over 48,000 concepts[1] which are connected by more than 61,000 IS-A links and 71,000 non-hierarchical (i.e., non-IS-A) relationships. The figures for the InterMED are as follows: There are approximately 2,500 concepts, 3,400 IS-A links, 3,500 non-hierarchical relationships.

Both the MED and InterMED obey the following rule pertaining to the introduction of properties into the vocabulary.

Rule (Uniqueness of Property Introduction): A given property x can be introduced at only one concept in the CV.

If other concepts also need x, then they must be defined as descendants of the concept at which x is introduced and obtain it by inheritance. We will be assuming that any CV to which our methodology is to be applied satisfies the above rule. Note that this is not overly restrictive because if there is a need to introduce a property x at several independent concepts, then an "artificial" superconcept of these concepts can be created for the purpose of defining x [11].

[1] Currently the MED has over 56,000 concepts and the InterMED has 2,820 concepts. At the time the work reported in this chapter was done, the MED had around 48,000 concepts.

2.2 Modeling Alternatives

From our point of view, there are three major alternatives for the solution of the vocabulary modeling problem. Simply stated, one can either: (1) Define a single object class and make all concepts instances of it; (2) Define a class for each concept and forgo instances; or (3) Choose a middle-ground between (1) and (2), that is, define a number of object classes and make each concept an instance of the "appropriate" one. Our methodology is based on (3). Before getting to it, we will examine (1) and (2) and discuss why they were rejected.

Because a CV is a collection of concept nodes (and links between them), one approach to building an OODB schema for it is to define a single object class, say, *Concept* and make all the nodes instances of it. Properties common to all concepts, such as *name* and *synonyms*, would be defined at *Concept*. Other properties defined for particular concepts and their descendants would be modeled as separate objects, instances of another class called *Property*. To connect a concept to its entire set of properties, a pair of converse relationships, *has-property* and *is-property-of*, would be maintained between *Concept* and *Property*. As an example, consider the concept **Glucose** (Figure 2.1). In this arrangement, there would exist an instance of *Concept* representing **Glucose**. Its attribute *name* would have the value "Glucose." The concept **Glucose** exhibits the relationship *is-measured-by*, for which the target concept is **Glucose Test**. An instance of *Property* is created to represent this link, and this instance is associated with **Glucose** via the converse relationship pair, *has-property* and *is-property-of*.

The three objects—two instances of *Concept* and one instance of *Property*—and their interconnections are shown graphically in Figure 2.1. Above the dashed line are classes *Concept* and *Property*. We use the following graphical conventions when drawing the OODB classes. A class is a rectangle with its name written inside. The names of any attributes defined by the class are written below the class's name and

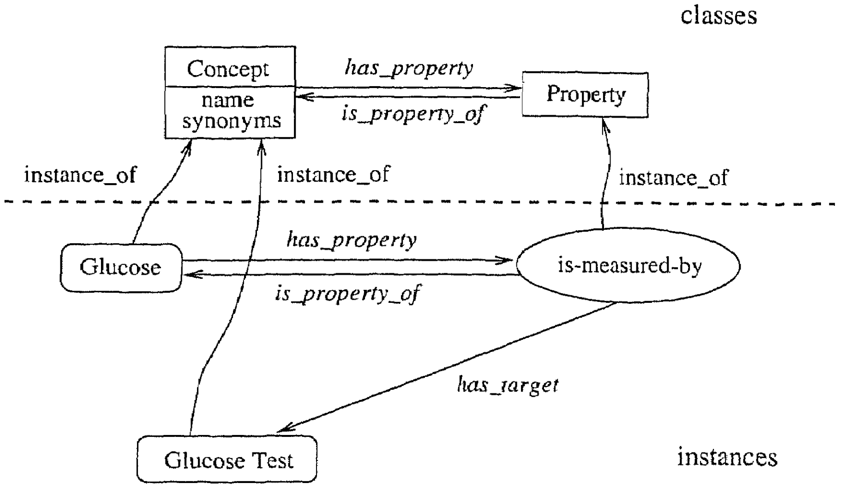


Figure 2.1 Object representation of the relationship *is-measured-by* between **Glucose** and **Glucose Test**

are separated from it by a line. A relationship is a labeled (in italics) arrow directed from the source class to the target class. Below the dashed line are three instances of classes *Concept* and *Property*. That indicates concept **Glucose** *is-measured-by* **Glucose Test.**

An advantage of this modeling alternative is that a property of a concept cannot be assigned a value unless the property is actually defined for the concept. However, this alternative does have a number of drawbacks. First, in order to access or manipulate a concept in its entirety, it would have to be "joined" together from its constituent objects (one instance of *Concept*, and potentially many instances of *Property*). The second is a proliferation of database objects resulting from the fact that most properties would be objects themselves. For example, the MED has about 70,000 relationships that would be instances of *Property*. Another shortcoming is the "flattening" of the CV's IS-A hierarchy (due to the definition of a single class *Concept*) and the failure to exploit any property inheritance, a fundamental aspect of object-oriented modeling.

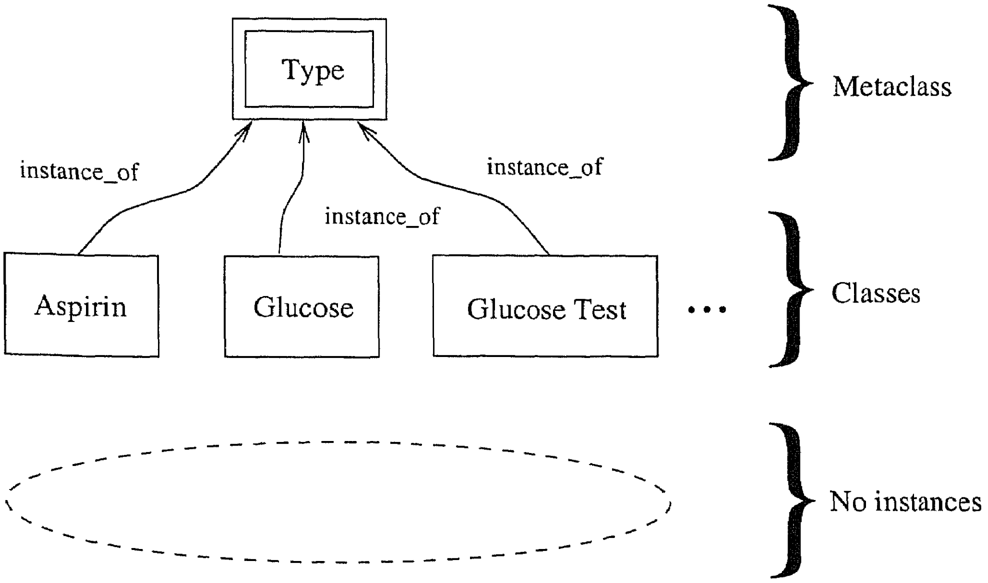


Figure 2.2 Concepts as classes, i.e., instances of the metaclass *Type*

The second alternative representation arises from the view of the CV as a strictly intensional entity. Each concept can be seen as a description of a general category or *class* in the linguistic sense. For example, **Aspirin** denotes the general concept of a kind of drug (namely, Aspirin), not some specific brand tablets or preparations composed of it. With this in mind, we could represent each concept as its own object class if the target OODB system supports some kind of run-time manifestation of classes in a metaschema or data dictionary form. The ONTOS DB/Explorer, our implementation vehicle, indeed supports such a feature. Note that under this arrangement the OOVR would consist solely of classes (represented, presumably, as objects in the metaschema), but it would have no instances of those classes. Figure 2.2 shows the concepts **Aspirin**, **Glucose**, **Glucose Test**, and so on, as instances of the metaclass *Type* (drawn as a double-edged box) in an ONTOS metaschema. The concepts here are drawn as boxes to indicate their status as classes.

There are two major shortcomings to this approach. First, the OODB schema would be enormous, and instead of aiding a user in the comprehension of the contents of the CV—as would be expected—it would be entirely overwhelming and incomprehensible. The second problem has to do with the representation of the concepts' properties and their associated values. In ONTOS, for example, the properties of a class are stored as independent objects apart from the run-time manifestation of their class within the metaschema. Because these "property" objects are strictly intensional, there is no provision for associating actual values with them as required by the CV's structure. To overcome this problem, we could augment the metaschema with additional constructs to associate a property with values for the different concepts that possess it. (As discussed, a property defined at one concept appears at all its descendants, too, via inheritance. Each of the descendents would have its own value for the property in question.) Alternatively, many different property objects could be used to make the connections between the concepts and their values for the given property. This would result in the need for a great deal of extra storage within the metaschema, and would also require its reorganization. Such practices, while feasible, are not practical. Additionally, access to metaschema objects is typically not optimized for frequent usage. Thus, we reject both of these extreme solutions, and we now turn to the approach that we have adopted.

2.3 OODB Schema for CVs without Intersection Concepts

In the two alternative approaches described in the previous section, either the OODB representation of the CV consisted only of instances (of a single class) or only of classes. Our methodology is situated in the range between these two extremes with the schema having more than one class but nowhere near an amount equal to the number of concepts in the source CV. The ultimate task is to determine how many classes are necessary, what the classes look like, what their interrelationships should, and to which classes the various concepts should belong.

The OODB schema produced by our approach is derived automatically from an overall structural analysis of the CV. It is based on the partitioning of the CV into groups of concepts that exhibit the exact same set of properties. To be more precise, we will need the following definition, where we use $\mathrm{P}(\mathbf{x})$ to denote the entire set of properties of the concept $\mathbf{x}$.

Definition 1: **(Area)** Let A be a non-empty set of concepts such that $\forall \mathbf{x}, \mathbf{y} \in A, \mathrm{P}(\mathbf{x}) = \mathrm{P}(\mathbf{y})$. For such a set, let $\mathrm{P}(A) = \mathrm{P}(\mathbf{x})$ for some $\mathbf{x} \in A$. That is, $P(A)$ is the set of properties exhibited by each of A's members. A is called an *area* if there does not exist a set of concepts B such that $\forall \mathbf{v}, \mathbf{w} \in B, \mathrm{P}(\mathbf{v}) = \mathrm{P}(\mathbf{w}), \mathrm{P}(B) = \mathrm{P}(A)$, and $A \subset B$. In other words, A is an area if it is the maximal set of concepts which exhibit the set of properties $\mathrm{P}(A)$. □

By definition, if A_1 and A_2 are areas, and $A_1 \neq A_2$, then $A_1 \cap A_2 = \emptyset$. That is, distinct areas are always disjoint. Given this fact, it is a relatively straightforward task to define the OODB schema of a CV if one knows all the areas of the CV. For each area A, a class having the properties $\mathrm{P}(A)$ is created. While this description leaves out a few important details, it does capture the essence of the approach. The real problem, of course, lies in the identification of the areas.

In the remainder of this section, we will be describing the process of identifying all areas in the CV—and, hence, partitioning the CV into mutually exclusive sets—under the assumption that the CV does not contain *intersection concepts*, which will be formally defined in the next section. Informally, an intersection concept is one that does not introduce any new properties but still has a different set of properties than all its parents due to inheritance from several of them. After discussing the area-partitioning, we will present the details of the OODB schema derived directly from it and describe how the CV is stored in the database.

Let us point out that the methodology as presented in this section is sufficient in itself for a CV whose IS-A hierarchy is a tree. Examples of such CVs include ICD-9 [106] and AHFS [2]. Even if the hierarchy is a DAG, the methodology will still be suitable if the CV is devoid of intersection concepts, a condition that can be detected algorithmically. An example of such a CV is NDC [81]. The main reason for delaying consideration of intersection concepts to the next section is that it greatly simplifies the presentation.

The identification of areas follows the pattern in which the concepts' properties are introduced into the CV. In this regard, we will need the following two definitions. In the first, we use $\Pi(\mathbf{v})$ to denote the set of properties intrinsically introduced or defined by the concept **v** (as opposed to those that are inherited by it).

Definition 2: (Property-Introducing Concept) A concept **v** is called a property-introducing concept if $\Pi(\mathbf{v}) \neq \emptyset$, i.e., if it intrinsically defines one or more properties. □

Definition 3: (Direct Property-Introducing Descendant [DPID]) Let **v** and **w** be property-introducing concepts, and let **w** be a descendant of **v** with respect to the IS-A hierarchy. The concept **w** is called a direct property-introducing descendant (DPID) of **v** if there exists an upwardly directed IS-A path (there can be more than one) from **w** to **v** that does not contain another property-introducing concept. □

The property-introducing concepts form the basis for the areas of the CV. In fact, in a CV that satisfies the condition regarding the absence of intersection concepts, we can equivalently state the definition of area in terms of property-introducing concept as:

Definition 4: (Area [equivalent redefinition]) An area is a set of concepts containing a property-introducing concept **v** and all of **v**'s descendants excluding its DPIDs and their respective descendants. □

Clearly, an area can contain only a single property-introducing concept. Any descendants that are also property-introducing concepts will define new areas of their own. From a top-down vantage point, the property-introducing concept is the highest node in an area, and in this sense it "starts" the area. For this reason, we refer to that concept as the root of the area and use it when we need to assign a name to the area.

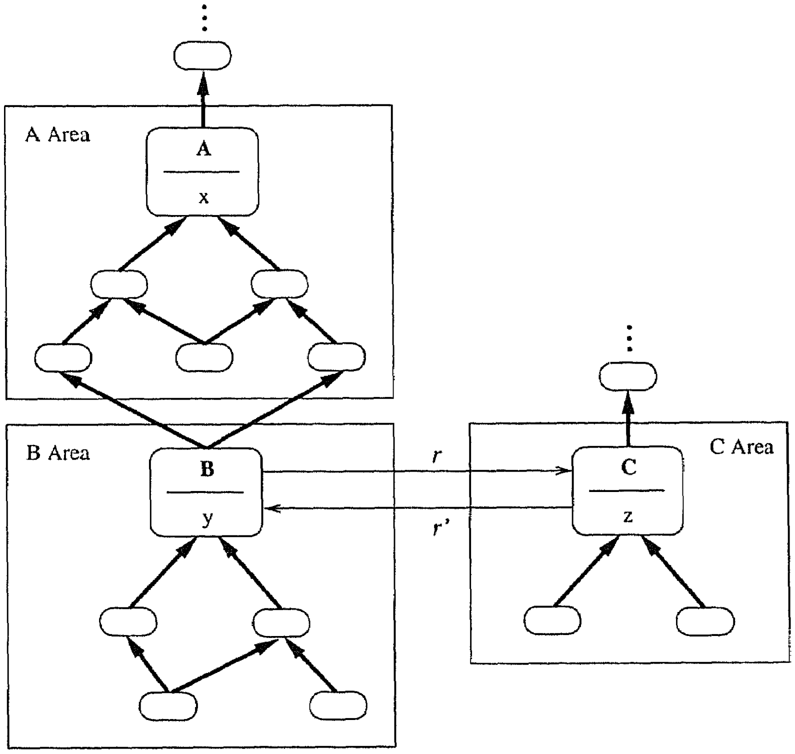


Figure 2.3 Three areas of a CV

To illustrate the partitioning of a CV, we show three areas A, B, and C in Figure 2.3. The concepts are drawn as rectangles with rounded edges, while the areas are shown as large rectangles. Note that the root of area A (i.e., the concept **A**) introduces the single attribute x. Area A extends down to, but excludes, concept

B which is a DPID of **A**. **B** defines the attribute y as well as the relationship r and serves as the root of area B. Finally, area C has the root **C** which introduces attribute z and the relationship r', the converse of r. The ellipses in the figure indicate the omission of additional concepts above the areas A and C.

As a concrete example from the InterMED, the concept **Measurable Substance** introduces a new relationship *measured-by* and is thus the root of a new area, "Measurable Substance" area. Examples of concepts in that area are **Color**, **Temperature**, **Specific Gravity**, **Viscosity**, **Blood Coagulation**, and **Optical Density**. Another example is the area rooted at the concept **Entity**, which, as we noted above, all CVs are assumed to be rooted at overall. **Entity** introduces a number of properties (including *name*) and is therefore the root of "Entity" area.

Once the areas of the CV have been identified, the OODB schema can be created as follows. For each area A, define a class (called *A_Area*) whose direct instances will be exactly the concepts in A, including A's root (call it $\mathbf{r}_A$). As all concepts in A possess the exact set of properties as $\mathbf{r}_A$ [namely, $\mathrm{P}(\mathbf{r}_A)$], it seems natural to define the properties of *A_Area* to be $\mathrm{P}(\mathbf{r}_A)$. However, this ignores the fact that $\mathbf{r}_A$ may have inherited some of its properties rather than having defined them all intrinsically. In fact, it is only necessary to endow *A_Area* with the set of intrinsic properties of $\mathbf{r}_A$, $\Pi(\mathbf{r}_A)$. The rest of the properties can be obtained via inheritance in a way which mimics the inheritance taking place in the semantic network. The root $\mathbf{r}_A$ inherits from its parents—which reside in areas distinct from A—those properties that it does not itself define. From this, it can be seen that *A_Area* should inherit from all the classes which represent areas that contain a parent of $\mathbf{r}_A$. That is, *A_Area* should be a subclass of all those classes. To characterize this precisely, let us define the following.

Definition 5: (Parent [Area]) Let B and C be areas. If a parent of r_C, the root of C, resides in B, then B is called a parent (area) of C. □

Continuing, let the areas $T_1, T_2, \ldots, T_n$ $(n \geq 1)$ be the parents of area A. In other words, $T_1, T_2, \ldots, T_n$ each contain at least one parent concept of $\mathbf{r}_A$. It will be noted that $\mathrm{P}(A) = \Pi(\mathbf{r}_A) \cup \mathrm{P}(T_1) \cup \mathrm{P}(T_2) \cup \cdots \cup \mathrm{P}(T_n)$. That is, the properties of the area A are gathered from its root and all its parents. To capture this in the schema, *A_Area* is defined with the set of intrinsic properties $\Pi(\mathbf{r}_A)$, and as a subclass of all n classes T_1*_Area*, T_2*_Area*, through T_n*_Area* representing, respectively, the areas $T_1, T_2, \ldots, T_n$. Let us note that it is possible that one of these classes, say, T_i*_Area* is a parent or an ancestor of another, say, T_j*_Area*. In such a case, defining subclass relationships between *A_Area* and both T_i*_Area* and T_j*_Area* would lead to a "short cut" (i.e., a materialization of a transitive subclass connection) in the OODB schema. The relationship between *A_Area* and T_i*_Area* is clearly redundant because, in such a situation, $\mathrm{P}(T_i) \subset \mathrm{P}(T_j)$. Therefore, the subclass relationship to T_j*_Area* gives *A_Area* all the properties that the relationship with T_i*_Area* would provide. Due to this, the subclass link between *A_Area* and T_i*_Area* is omitted.

Since all concepts (except for **Entity**) have superconcepts, $\mathbf{r}_A$'s parents will probably have their own parents, and so the subclass relationships of the schema will branch upward in a DAG structure until they reach the class *Entity_Area* representing the top area (i.e., "Entity" area) of the CV.

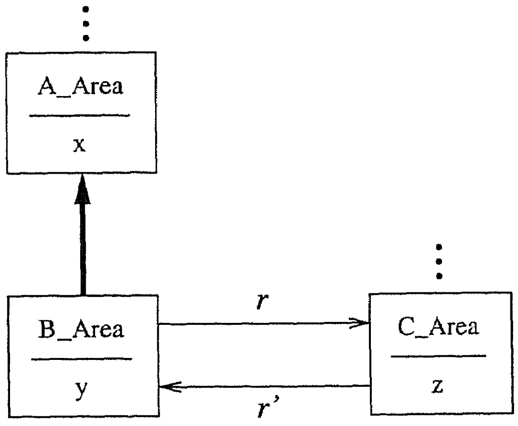


Figure 2.4 Area classes corresponding to the three areas in Figure 2.3

In Figure 2.4, we show the three classes that represent the areas from Figure 2.3. A class is drawn as a rectangle; its intrinsic attributes are listed inside beneath its name. A subclass relationship is denoted as a bold arrow directed upward from the subclass to its superclass, and a relationship is represented by a labeled thin arrow. As we see, *A_Area* has the attribute x defined by the concept **A**, the root of the area. Likewise, *B_Area* has the attribute y and the relationship r, and *C_Area* has the attribute z and the relationship r'. Note that *B_Area* is a subclass of *A_Area* because area A contains the parents of concept **B**, the root of area B.

Referring back to the sample areas from the InterMED, "Measurable Substance" area would have the corresponding class *Measurable_Substance_Area*, including the relationship *measured-by*, among others. "Entity" area would be associated with the class *Entity_Area* possessing the property *name* and so forth. It should be noted that *Entity_Area* is the root class of the schema.

One final aspect of the OODB that warrants special consideration is the representation of the CV's IS-A hierarchy. We have already used this feature and, more specifically, its inheritance mechanism to derive the subclass relationships of the schema. However, it is still required that the individual concepts themselves (at the instance-level of the OODB) be connected to their parents (and vice versa). This can be done by once again noting that all concepts in the CV, except for the root, have parents. Thus, we equip all concepts with two additional generic relationships, "has-superconcept" and "has-subconcept," which connect a concept to its parents and children, respectively. Within the semantic network, these relationships can be seen as being defined by **Entity** and therefore inherited by every other concept. Following that, they are defined reflexively at the root class *Entity_Area* of the OODB schema. If, in the original CV, concept **v** IS-A **w**, then, in the OODB, the object representing **w** is a referent of **v** with respect to the *has_superconcept* relationship; conversely, **v** is a referent of **w** via *has_subconcept*.

Because the direct extension of a class in the OODB schema is precisely one area, we refer to it as an *area class.* Overall, the schema comprises a collection of area classes. Since it is an abstraction of the property definitions and accompanying inheritance that occur within a CV as modeled by a semantic network, we call this kind of schema a *network abstraction schema.*

It is important to point out that the area-partitioning described above does not lead to a proliferation of classes in the OODB schema. There are two major reasons for this: (a) Typically, there is a small number of properties in a CV compared to the total number of concepts; and (b) The "uniqueness of property introduction" rule. Due to the latter, one does not find redundant property introductions strewn throughout the CV. Since property-introducing concepts always start new areas, this helps keep their numbers down.

Point (a) is, fortunately, a general characteristic of CVs which stems from the fact that they are definitional structures rather than dynamic data stores. In the InterMED,[2] there are only 51 distinct properties for a total of about 2,500 concepts. For the MED, the number is 150 properties for approximately 48,000 concepts. As a consequence of this, very few concepts intrinsically introduce properties; most properties are inherited. There are just 26 property-introducing concepts in the InterMED and 57 in the MED. It is interesting to contrast this sparseness of property introduction in a CV with the denseness of property introduction in a typical OODB schema where at (almost) every class we expect to find the definitions of new properties.

In Figure 2.5, we show the OODB schema for the InterMED in the case where its intersection concepts (and their descendants) are omitted. It should be noted that the subclass hierarchy of such a schema does not necessarily have a tree structure

[2] As of Fall 1996

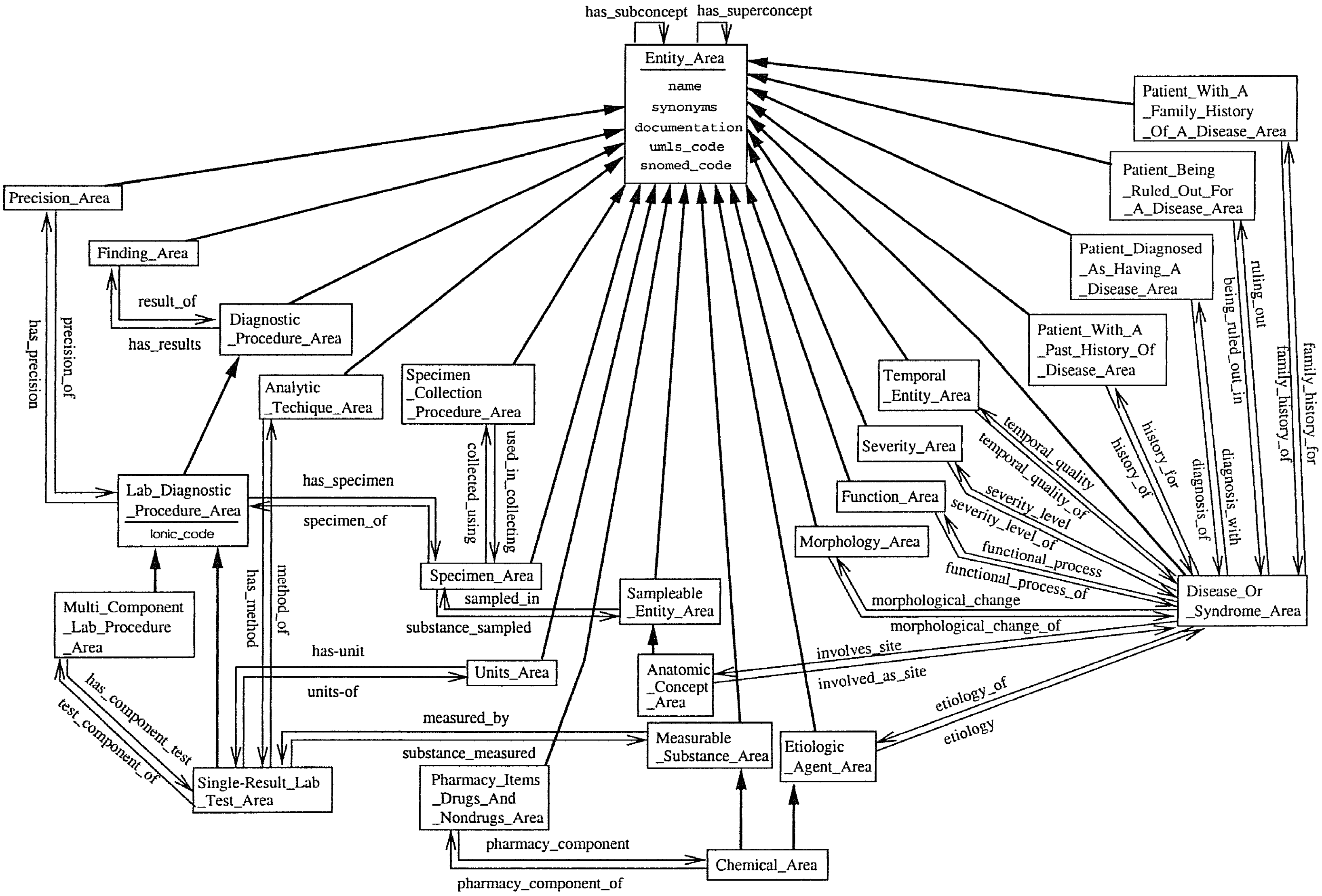


Figure 2.5 Schema for the InterMED excluding intersection concepts

because a property-introducing concept can have parents in many different areas. The concept **Chemical** is an example. See the class *Chemical_Area* in the figure.

2.4 CVs with Intersection Concepts

The problem of identifying areas is made more difficult when intersection concepts are present in the CV. Before formally defining what we mean by this notion, let us give an illustration. In Figure 2.6, we show a more complex version of the CV excerpt appearing in Figure 2.3. In the following, we will call two concepts *ancestrally related* if there exists an ancestor/descendant relationship between them.

According to our specification of an area in terms of a property-introducing concept given in the previous section, the concepts **D**, **E**, **F**, and **G** (enclosed in a large box) should belong to the area rooted at **B** (i.e., area B) since they are "between" it and one of its DPIDs, namely, **H**. However, on closer inspection, they similarly belong to the area C. On the other hand, those concepts cannot belong to area B [area C] since they have extra properties not in P(B) [P(C)] which they inherit from C [B]. In fact, they make up a new area of their own, even though none is a property-introducing concept. They obtain their properties via inheritance from two other areas that are, in a sense, independent. Each of the concepts **D** and **E** can be seen to lie at the juncture of some inheritance paths emanating downward from the ancestrally unrelated property-introducing concepts **B** and **C**. For this reason, we call **D** and **E** *intersection concepts.* While we could formalize this notion in terms of IS-A paths, it is simpler to do it as follows.

Definition 6: (Intersection Concept) Let **x** be a concept which is not a property-introducing concept and which has multiple superconcepts $\mathbf{t}_1, \mathbf{t}_2, \ldots, \mathbf{t}_n$ $(n > 1)$. Then **x** is called an intersection concept if it satisfies the following: $\forall i : 1 \leq i \leq n$, $\mathrm{P}(\mathbf{x}) \neq \mathrm{P}(\mathbf{t}_i)$. In other words, the set of properties of **x** differs from the set of properties of each of **x**'s parents. Note that $\mathrm{P}(\mathbf{x}) = \bigcup_{i=1}^{n} \mathrm{P}(\mathbf{t}_i)$. □

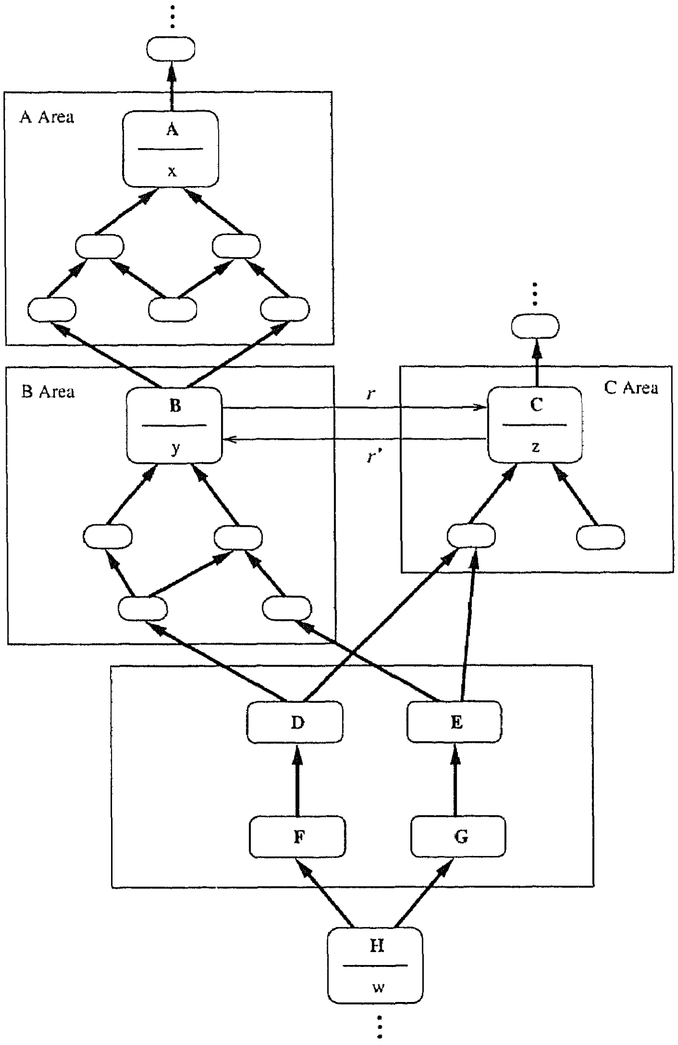


Figure 2.6 A more complex version of Figure 2.3 containing intersection concepts **D** and **E**

Note that a concept having a single parent cannot be an intersection concept: Its properties could only differ from its parent's if it intrinsically introduced some, in which case it would be a property-introducing concept. Furthermore, at least two of its parents must be from different areas. Another characteristic of intersection concepts, that has bearing on the area-partitioning of the CV, is that two such concepts having identical sets of properties (e.g., **D** and **E**) cannot be ancestrally related.

For CVs containing intersection concepts—e.g., the MED and the InterMED—there are two different kinds of areas. The first, discussed in the previous section, starts at a single property-introducing concept and extends downward until other property-introducing concepts or intersection concepts are reached. The second kind, defined below, is rooted in one or more intersection concepts and branches down in an identical manner to that of the first kind. We will call the first kind of area a *property-introducing area*; the second will be referred to as an *intersection area.* To define these more precisely, we will need the following.

Definition 7: (Direct Intersection Descendant [DID]) Let **v** be a property-introducing concept, **w** be an intersection concept, and let **w** be a descendant of **v** with respect to the IS-A hierarchy. The concept **w** is called a direct intersection descendant (DID) of **v** if there exists an upwardly directed IS-A path (there can be more than one) from **w** to **v** that does not contain another property-introducing concept or intersection concept. □

In Definitions 3 and 7, we use a property-introducing concept **v** as the ancestor with respect to which DPID and DID are defined. It is a straightforward matter to define both DPID and DID with respect to an intersection concept ancestor, as well. We omit these two additional definitions for the sake of brevity. We will, however, be utilizing them and referring to an intersection concept's DPIDs and DIDs below.

Definition 8: (Property-introduction Area) A property-introducing area is a set of concepts containing a property-introducing concept **v** and all of **v**'s descendants excluding its DPIDs and DIDs and their respective descendants. □

Definition 9: (Intersection Area) Let $E = \{\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_n\}$ be a set of intersection concepts (as defined in Definition 6) such that $\forall i,j : 1 \leq i,j \leq n$, $\mathrm{P}(\mathbf{e}_i) = \mathrm{P}(\mathbf{e}_j)$.

Furthermore, let E be maximal, i.e., there does not exist an intersection concept $\mathrm{s} \notin E$ such that $\mathrm{P}(\mathbf{s}) = \mathrm{P}(\mathbf{e}_k)$ for some k. Now, $\forall i : 1 \leq i \leq n$, let E_i be the set containing $\mathbf{e}_i$ and all of $\mathbf{e}_i$'s descendants excluding its DPIDs and DIDs and their respective descendants. The set $I = E_1 \cup E_2 \cup \cdots \cup E_n$ is called an intersection area. □

The concepts $\mathbf{e}_1$, $\mathbf{e}_2, \ldots$, $\mathbf{e}_n$ will be called the roots of the intersection area because each starts a portion of it. These portions may overlap. The name of the area can be chosen arbitrarily (perhaps by the vocabulary administrator) from among the n concepts. In Section 5.1, we will discuss possible naming conventions.

Referring back to Figure 2.6, we see that the four concepts, **D**, **E**, **F**, and **G**, constitute an intersection area. Both **D** and **E** are intersection concepts and serve as the roots of the area. The concepts **F** and **G** are not roots. Indeed, they are not intersection concepts but happen to reside in an intersection area by dint of their IS-A connections to intersection concepts.

In the InterMED, only 2 of its 2,500 concepts are intersection concepts. For the MED, it is 1,332 out of 48,000. An example of an intersection concept in the InterMED is **Water** whose parents reside in two areas: "Sampleable Entity" area and "Chemical" area. An intersection concept from the MED is **Chloramphenicol Preparations** whose parents belong to three areas, "Antihistamine Drugs," "Drug Allergy Class," and "DEA Controlled Substance Category."

In the OODB schema, property-introducing areas are treated in the same manner asas described previously. One *property-introducing* (*area*) *class* is created for each property-introducing area. The properties and subclass relationships of the class are determined by the area's root and its parents, respectively.

For an intersection area, a class [called an *intersection* (*area*) *class*] is defined as we previously defined a property-introducing class for each property-introducing area. However, this class contains *no* intrinsic properties. Instead, it gets all its properties via inheritance—just as its roots do.

The subclass relationships originating at an intersection class are once again determined by the parents of a root. A subtlety that arises here comes from the fact that the parents of one root may reside in areas different from those of the parents of another root. Even so, the union of the parents' sets of properties with respect to one root is always the same as the union with respect to any other. If not, the roots would belong to different areas.

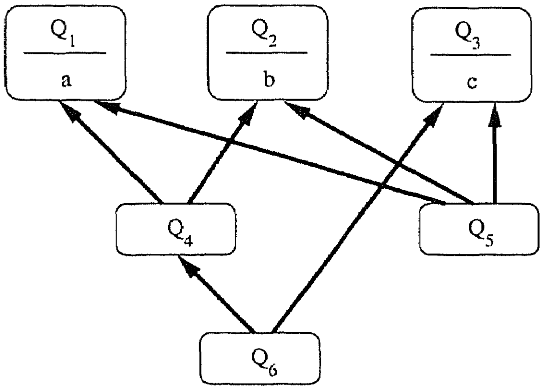


Figure 2.7 Parents of the roots of an intersection area residing in different areas

To illustrate this point, consider the six concepts, Q_1 through Q_6, shown in Figure 2.7. Concepts Q_1, Q_2, and Q_3 introduce the attributes a, b, and c, respectively, and therefore serve as the roots of three different property-introducing areas. (In this simplified configuration, they are the only concepts in their respective areas.) Q_4 is an intersection concept possessing the properties a and b obtained via inheritance from its parents. The interesting aspect of the figure involves the concepts Q_5 and Q_6. Both are intersection concepts and possess all three attributes, a, b, and c. Therefore, they are roots of the same intersection area. However, Q_5 has the three parents Q_1 Q_2 and Q_3 residing in their own property-introducing areas. Concept Q_6 shares the parent Q_3 with Q_5 but has only one other parent Q_4 which, as noted, is an intersection concept. To summarize, we see that the sets of parent areas of an intersection area are not unique. They can differ with respect to its various roots.

Because of this, there are potentially many equivalent subclass configurations (and, hence, OODB schemas) that can be used to represent such an intersection area. The question is: Which of these should be chosen? Our answer is to select the root whose parents collectively reside in the fewest areas and define the subclass relationships with respect to those area classes. This minimizes the required number of subclass relationships.

To demonstrate this, let us refer back to Figure 2.7. We would select the relationships of the root $\mathbf{Q}_6$ for the intersection area containing both $\mathbf{Q}_5$ and $\mathbf{Q}_6$ because $\mathbf{Q}_6$'s parents reside in two areas while $\mathbf{Q}_5$'s reside in three. The subclass relationships for this intersection area class would be directed to two classes, one representing the property-introducing area of concept $\mathbf{Q}_3$ and the other representing the intersection area rooted at $\mathbf{Q}_4$.

In general, the process of determining the subclass relationships for an intersection class is as follows. Let I be an intersection area and let $\mathbf{r}_I$ be one of its roots whose parents reside in the fewest different areas.[3] Moreover, let $T_1, T_2, \ldots, T_n$ be all the areas containing at least one of $\mathbf{r}_I$'s parents. Then the class *I_Area*, the intersection class for I, is defined as a subclass of T_1*_Area*, T_2*_Area*, through T_n*_Area*, the respective area classes of $T_1, T_2, \ldots, T_n$.

As pointed out above, an intersection concept's parents must reside in at least two different areas, so an intersection class will have at least two superclasses. This demonstrates that the OODB schema, for this type of CV, will exhibit multiple inheritance.

To illustrate the schema construction, we show the area classes for the areas from Figure 2.6 in Figure 2.8. The ellipses indicate the omission of subclass

[3] There may be more than one such root. In that case, the choice is made arbitrarily.

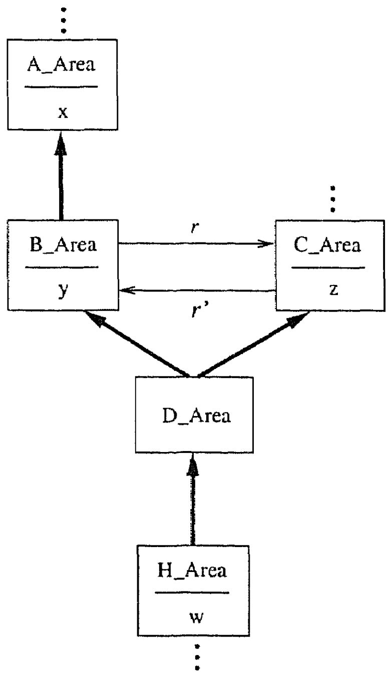


Figure 2.8 Area classes for the areas in Figure 2.6

relationships and additional classes that would appear in an expanded drawing. The three classes, *A_Area*, *B_Area*, and *C_Area*, are defined as discussed previously. Each is a property-introducing class. The class *D_Area* is an intersection class representing the intersection area containing the four concepts, **D**, **E**, **F**, and **G**. Both **D** and **E** are roots of the area and are thus viable designations for it. The name *D_Area* was chosen because **D** appears first in a scan of the area. The class does not have any intrinsic properties. It is a subclass of *B_Area* and *C_Area* because **D**'s parents (as well as **E**'s) belong to the areas B and C. The last class *H_Area* is a property-introducing class which introduces the attribute w and is a subclass of *D_Area*.

In Figure 2.9, we show the entire InterMED-OOVR schema comprising a total of 28 area classes and 30 subclass relationships. Of the 28 classes, 26 are property-introducing classes and 2 are intersection classes. One of the intersection classes is *Water_Area* which is a subclass of *Sampleable_Entity_Area* and *Chemical_Area*. The

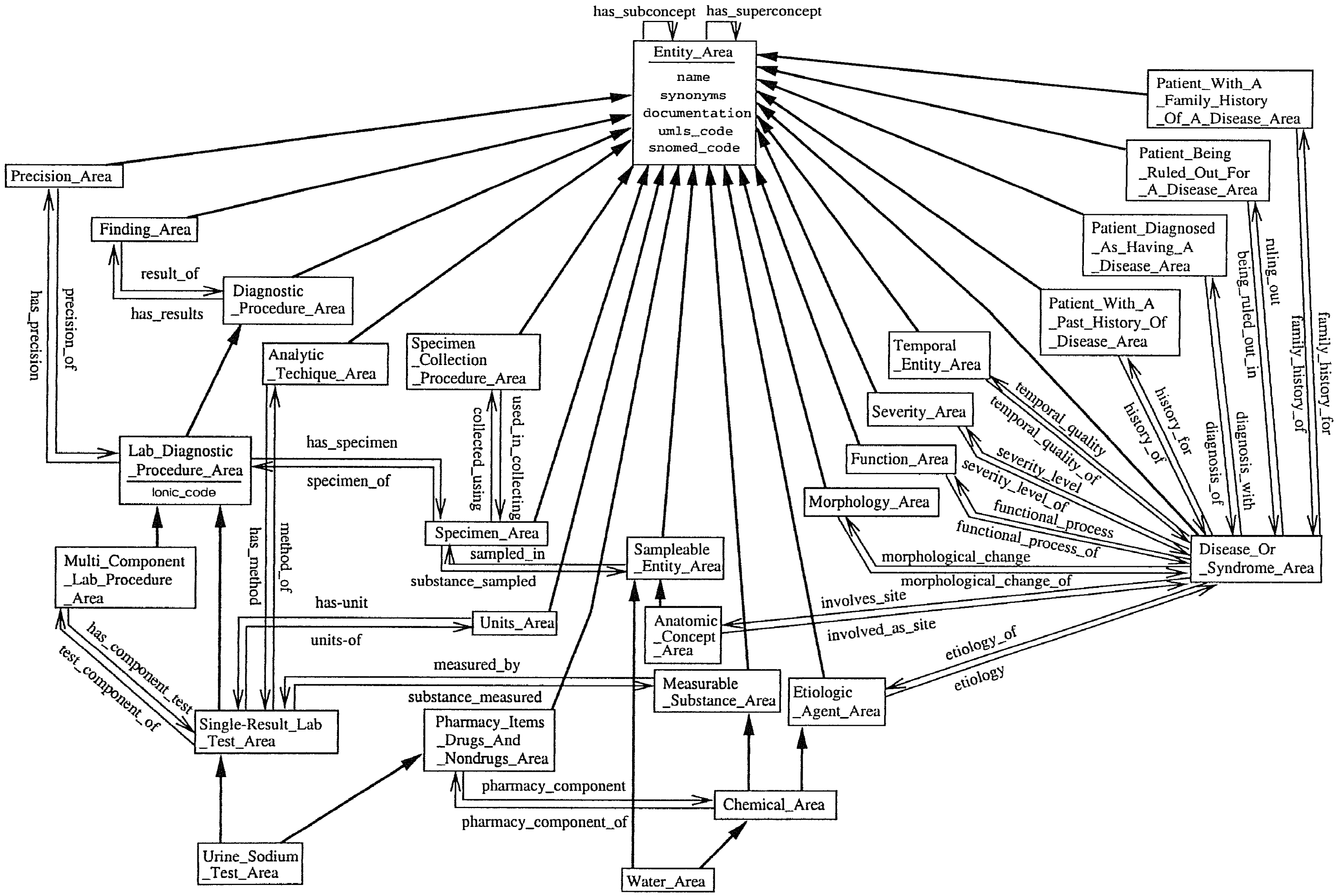


Figure 2.9 InterMED-OOVR schema

other is *Urine_Sodium_Test_Area* which has the parents *Single_Result_Lab_Test_Area* and *Pharmacy_Items_Drugs_and_Nondrugs_Area*.

The schema for the MED-OOVR contains 90 classes (57 property-introducing classes, 33 intersection classes) and 134 subclass relationships. Due to its large size, it is not convenient to show the entire schema graphically on one page. In Figure 2.10, we show only the 57 property-introducing classes. In order to save space, we have represented the property names as numbers. The corresponding property names can be found in Table 2.1. Figure 2.11 contains all the property-introducing classes (with all properties omitted) as well as the following six intersection classes: *Antihistamine_Drugs_Area*, *Chloramphenicol_Preparations_Area*, *Organism_Area*, *Wuchereria_Bancrofti_Area*, *Black_Piedra_Area*, and *Abnormal_Finding_in_Body_-Substance_Area*. It should be noted that it is possible for one intersection class to be a subclass of another intersection class. This is demonstrated by three of the intersection classes, *Chloramphenicol_Preparations_Area*, *Wuchereria_Bancrofti_Area*, and *Black_Piedra_Area*. Moreover, *Black_Piedra_Area* is two levels below the intersection class *Organism_Area*. Note also that *Chloramphenicol_Preparations_Area* has three parent classes.

An important aspect of our methodology is the compactness of the resultant OODB schema. For the InterMED, which contains about 2,500 concepts, the schema has merely 28 area classes—about an 80-to-1 reduction. The MED contains approximately 48,000 concepts and has a schema of around 90 classes—about a 500-to-1 ratio! Additionally, we find a slow growth rate for the schemas with respect to the size of the source CVs. The content of the MED is nineteen times larger than that of the InterMED, yet its schema is only about three times the size.

In [39, 41], we showed how the compactness of the schema helped a vocabulary administrator uncover mistakes that had been introduced into the MED. We also discussed how this representation can be used as a tool for comprehending the content

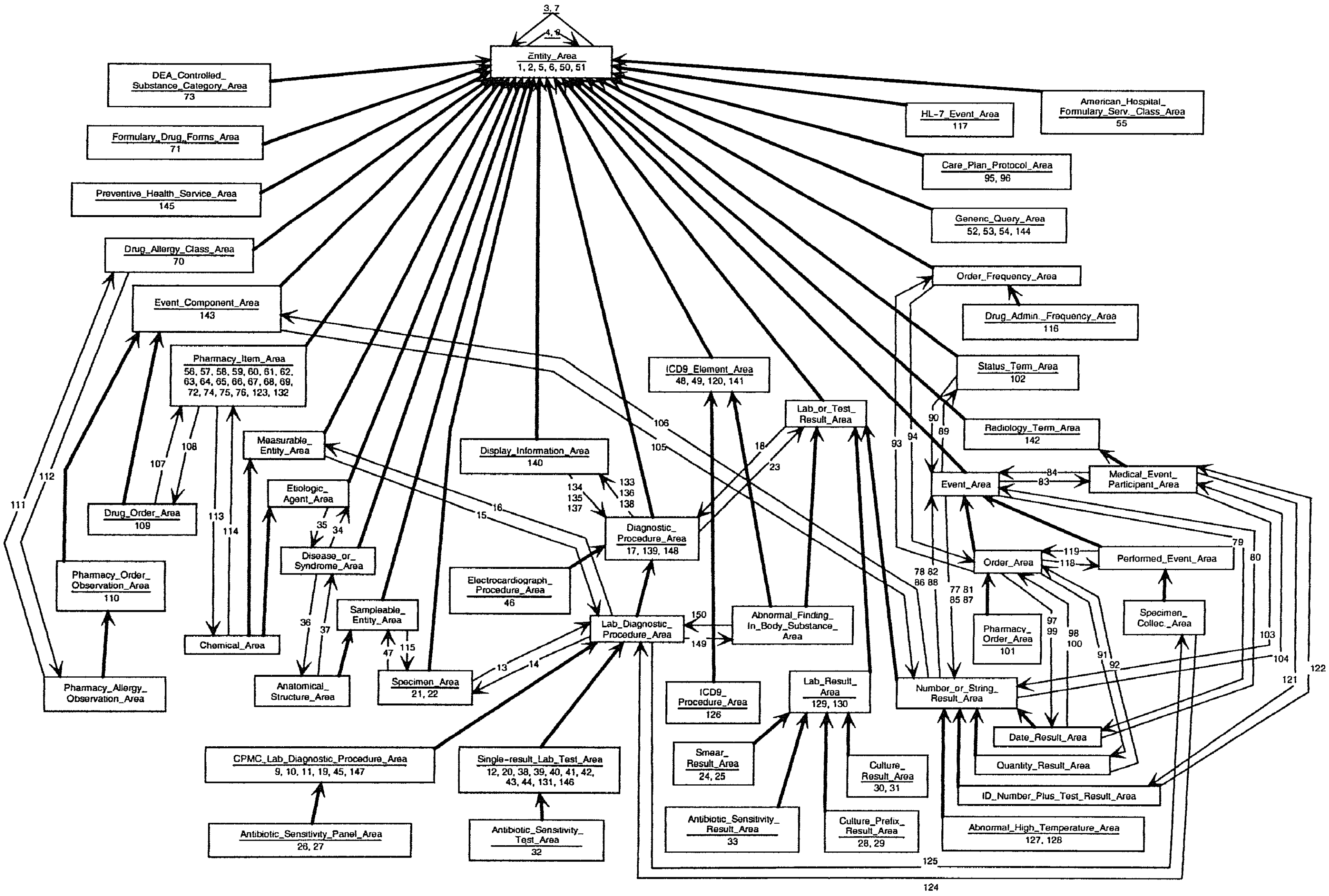


Figure 2.10 Property-introducing classes of MED-OOVR schema

Table 2.1 Names of properties in Figure 2.10

1	umls-code	2	name	3	has-subconcept
4	has-superconcept	5	synonyms	6	print-name
7	has-parts	8	part-of	9	cpmc-lab-proc-code
10	service-code	11	cpmc-unit-names	12	cpmc-lab-test-names
13	specimen-of	14	specimen	15	measured-by
16	substance-measured	17	units	18	result-of-tests
19	cpmc-lab-proc-name	20	cpmc-lab-test-code	21	cpmc-lab-spec-code
22	cpmc-lab-spec-name	23	result-type	24	cpmc-smear-code
25	cpmc-smear-name	26	cpmc-panel-code	27	cpmc-panel-name
28	cpmc-prefix-code	29	cpmc-prefix-name	30	cpmc-result-code
31	cpmc-result-name	32	cpmc-sensitivity-name	33	cpmc-sensitivity-result-name
34	etiology	35	causes-diseases	36	site
37	site-of-diseases	38	normal-value	39	low-normal-value
40	high-normal-value	41	male-low-normal-value	42	male-high-normal-value
43	female-low-normal-value	44	female-high-normal-value	45	normal-ranges-text
46	cpmc-ecg-name	47	substance-sampled	48	icd9-code
49	icd9-entry-code	50	main-mesh	51	supplementary-mesh
52	question-type	53	english-question	54	brs-question
55	ahfs-class-code	56	dose-strength-units	57	dose-strength-number
58	formulary-name	59	short-formulary-name	60	formulary-code
61	drug-trade-name	62	drug-generic-name	63	drug-manufacturer
64	drug-rx-vs-otc	65	drug-form-code	66	drug-floor-stock
67	drug-route	68	drug-in-formulary	69	drug-volume
70	allergy-class-code	71	drug-description	72	drug-category
73	dea-code	74	drug-specifier	75	drug-generic-code
76	drug-interaction-codes	77	event-id	78	event-id-of
79	event-date	80	event-date-of	81	event-patient-id
82	event-patient-id-of	83	event-participant	84	participant-of
85	event-organization	86	event-organization-of	87	event-location
88	event-location-of	89	event-status	90	status-of
91	order-quantity	92	order-quantity-of	93	order-frequency
94	order-frequency-of	95	protocol-name	96	protocol-short-name
97	order-start-date	98	order-start-date-of	99	order-stop-date
100	order-stop-date-of	101	pharmacy-order-code	102	status-code
103	participant-id	104	participant-id-of	105	order-value
106	order-value-of	107	ordered-drug	108	ordered-in
109	drug-role-code	110	pharmacy-observation-code	111	observed-allergy
112	allergy-observed-in	113	pharmaceutic-component	114	pharmaceutic-component-of
115	sampled-by	116	admin-frequency-abbrev	117	hl7-event-code
118	event-object	119	object-of-event	120	old-icd9-code
121	participant-name	122	participant-name-of	123	drug-id
124	collected-for	125	collected-by	126	cpt4-code
127	lower-limit-for-input	128	upper-limit-for-input	129	lab-message-code
130	lab-message-text	131	cpmc-long-test-name	132	drug-alert-code
133	has-default-displays	134	default-display-for	135	displays-elements-of
136	elements-displayed-by	137	has-display-parameters	138	is-display-parameter-of
139	has-test-display-class-name	140	display-parameter-order	141	icd9-name
142	cpmc-radiology-code	143	event-component-display-name	144	query-fillers
145	preventive-health-name	146	lab-alt-test-name	147	lab-alt-proc-name
148	has-proc-display-class-name	149	defined-by-test	150	defines-abnormal-finding

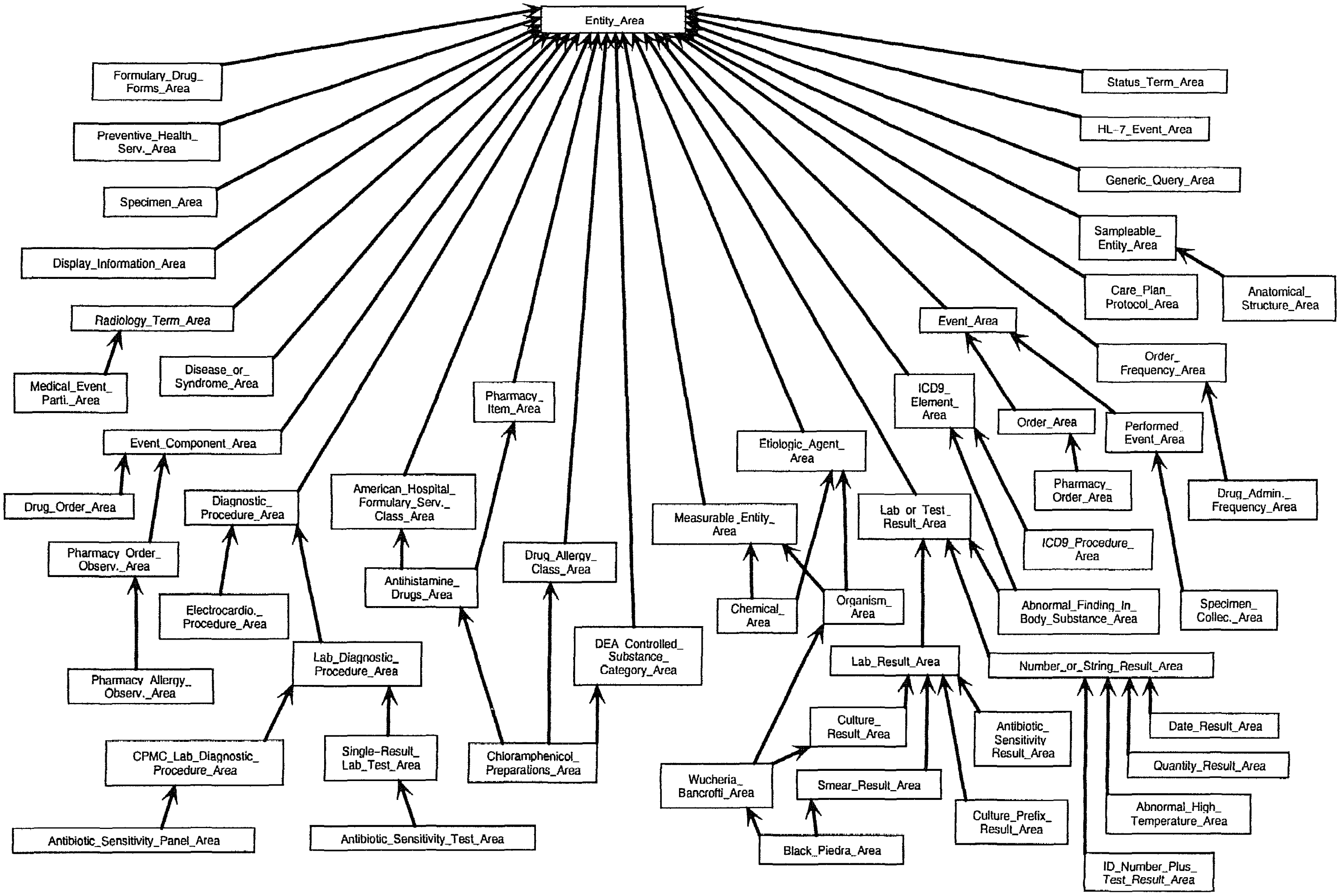

Entity_Area
Formulary_Drug_ Forms_Area
Preventive_Health_ Serv._Area
Specimen_Area
Display_Information_Area
Radiology_Term_Area
Medical_Event_ Parti._Area
Disease_or_ Syndrome_Area
Event_Component_Area
Drug_Order_Area
Pharmacy Order_ Observ._Area
Diagnostic_ Procedure_Area
Electrocardio._ Procedure_Area
Pharmacy Allergy_ Observ._Area
Lab_Diagnostic_ Procedure_Area
CPMC_Lab_Diagnostic_ Procedure_Area
Antibiotic_Sensitivity_Panel_Area
Single-Result_ Lab_Test_Area
Antibiotic_Sensitivity_Test_Area
American_Hospital_ Formulary_Serv._ Class_Area
Antihistamine_ Drugs_Area
Pharmacy_ Item_Area
Drug_Allergy_ Class_Area
Chloramphenicol_ Preparations_Area
DEA Controlled_ Substance_ Category_Area
Measurable_Entity_ Area
Chemical_ Area
Etiologic_Agent_ Area
Organism_ Area
Wucheria_ Bancrofti_Area
Black_Piedra_Area
Culture_ Result_Area
Smear_Result_Area
Lab_Result_Area
Lab or Test_ Result_Area
Culture_Prefix_ Result_Area
Antibiotic_ Sensitivity Result_Area
ICD9_ Element_ Area
ICD9_Procedure_ Area
Abnormal_Finding_In_ Body_Substance_Area
Number_or_String_Result_Area
Date_Result_Area
Quantity_Result_Area
Abnormal_High_ Temperature_Area
ID_Number_Plus_ Test_Result_Area
Event_Area
Order_Area
Pharmacy_ Order_Area
Performed_ Event_Area
Specimen_ Collec._Area
Order_ Frequency_Area
Drug_Admin._ Frequency_Area
Status_Term_Area
HL-7_Event_Area
Generic_Query_Area
Sampleable_ Entity_Area
Anatomical_ Structure_Area
Care_Plan_ Protocol_Area

Figure 2.11 Property-introducing and six intersection classes of MED-OOVR schema

of a CV. In fact, deriving an OODB schema for a CV was helpful for us both by the process and the result. The process of finding a schema gave us a source for asking intelligent questions about the vocabulary, the answers to which added insights to our comprehension of its knowledge content. The result, the schema itself, is a knowledge-rich abstraction that allows us to split the comprehension process into two steps. In the first step, a person studying the schema gets a good understanding of the CV's overall structure. In the second step, a person using the schema as a road map can then advance to studying selected areas of the CV in detail. In summary, a schema adds a valuable layer of abstraction on top of the large and complex content of a CV. We have built a program that utilizes this separation of CV and schema along with what we call "analogical forms" to provide an enhanced interface to CVs [31]. Using this program, a traversal of the CV can begin at the schema level and continue until the proper class is identified; at that point, the traversal can proceed at the concept level. Further abstraction of a CV can be achieved by partitioning the set of concepts of an area class into smaller units, as suggested in [43].

2.5 Program for Generating the OODB Representation of a CV

We have used our methodology to transform two existing medical CVs, the InterMED and the MED, into object-oriented representations. The methodology can be applied not only to medical CVs but to any semantic network-based vocabulary, as long as it satisfies the "uniqueness of property introduction" rule discussed earlier. Both OODB representations, called, respectively, the InterMED-OOVR and the MED-OOVR, are currently up and running on top of the ONTOS DB/Explorer OODB management system. The creation of each was done automatically by a program called the *OOVR Generator*, which can be used to convert any source CV into its equivalent OODB form.

In this section, we describe the overall architecture of the OOVR Generator. We will first present the assumed format of the source CV. We will then go on to discuss the components of the OOVR Generator's functionality.

2.5.1 Format of the Source CV

Various CVs can be assumed to be stored in different formats on disk. However, we will expect that a CV that is to be processed by our technique has a representation as a pair of text files, each having a specific syntax. If the desired source CV does not conform to this requirement, then it will first need to be converted. For this purpose, we include a *Preprocessor* module in the architecture of the OOVR Generator (see Figure 2.12). This portion may need to be modified for different CVs. To illustrate the necessary format, we will be referring to the InterMED, from which it was originally gleaned. The MED also employs this representation.

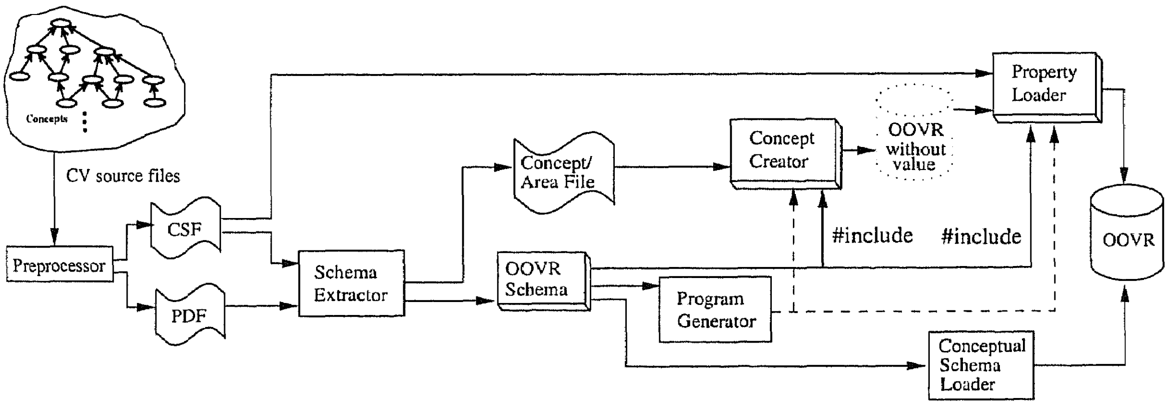


Figure 2.12 Architecture of the OOVR Generator

The two text files making up the disk-resident format of a CV are referred to as the *Property Definition File* (*PDF*) and the *Concept Specification File* (*CSF*). The PDF describes all the attributes and relationship types of the CV. Every attribute (or relationship type) is described by one line in the PDF. Each line is a triple whose first component is the property's number, which is assigned to a property in order to simplify references to it. The second component is the property's name; the third component is the number of the concept which introduces the property.

```
1, umls-code, 1
2, name, 1
3, descendant-of, 1
4, is-a, 1
5, synonyms, 1
6, print-name, 1
7, documentation, 1
8, snomed-code, 1
9, has-result, 3
10, result-of, 28
11, has-specimen, 4
12, specimen-of, 29
13, substance-measured, 5
14, measured-by, 30
15, has-precision, 5
```

(a) InterMED PDF

```
1,1,“T071”
1,2,“Entity”
3,2,“Diagnostic Procedure”
4,2,“Laboratory Diagnostic Procedure”
28,2,“Finding”
35,2,“Chemical Viewed Structurally”
37,2,“Inorganic Chemical”
37,4,35
37,8,“C-10090”
37,14,7
38,1,“T106”
38,2,“Element or Elemental Ion”
38,4,37
39,1,“C0037473”
39,2,“Sodium Ion”
```

(b) InterMED CSF

Figure 2.13 Excerpts of InterMED source files

There are 51 lines in the InterMED's PDF. Figure 2.13 (a) shows an excerpt of the file.[4] Note that the fields of a line are separated by commas. The MED's PDF contains 150 lines.

The second file, the CSF, describes all the details of the concepts' properties. Each line denotes the value for one property of some concept. A line in this file is also a triple. The first element is a concept number, uniquely identifying one of the concepts in the CV. The second number is a property number which stands for one of the relationship types or attributes and is therefore an index into the PDF. The third element may be another number (for a different concept) denoting the referent of a relationship. For an attribute, the third element is a primitive value, represented as a string type.

[4] The actual InterMED PDF contains some additional fields that are not relevant to the conversion process. These are removed by the Preprocessor.

The InterMED's CSF contains over 32,000 lines of text. Figure 2.13 (b) shows some of its entries. The line 37,2,"Inorganic Chemical" means that the concept 37 has the value "Inorganic Chemical" for the attribute *name* [property number 2 from Figure 2.13 (a)]. The entry 37,8,"C-10090" indicates that the SNOMED code of **Inorganic Chemical** is "C-10090." The line 37,4,35 means that the concept 37, **Inorganic Chemical**, has the "IS-A" relationship (property number 4) to concept 35, **Chemical Viewed Structurally**. The MED's CSF is quite a bit larger than that of the InterMED because the MED contains about nineteen times as many concepts having many more properties. In total, the MED's CSF has around 1,000,000 lines.

2.5.2 Architecture of the OOVR Generator

Figure 2.12 shows the overall architecture of the OOVR Generator. In the figure, we are using the following graphical conventions. A box represents a program module. A box with depth indicates that the module is generated by another module. The creation of such a module A by another module B is depicted by a dashed arrow from B to A. Ordinary arrows indicate the flow of data between modules either as files (wavy boxes) or databases (cylinders).

As we see from the figure, the OOVR Generator consists of five modules: *Preprocessor*, *Schema Extractor*, *Program Generator*, *Concept Creator*, and *Property Loader*. The Preprocessor is the only module which, as noted above, is CV-dependent. Hence, for a CV with a different file format than the InterMED, it would need to be modified.

The Schema Extractor is the first module to process the source CV. Its task is to carry out the area-partitioning and produce the appropriate OODB schema (as described in the previous section). It takes as its input both the PDF and the CSF. The output consists of two items, a "Concept/Area File" and the OODB schema for the OOVR. The Concept/Area File simply holds the mapping between the concepts of the CV and the areas derived from the partitioning process. Effectively, it is a two-column table, where the first column contains the concept names, and the second holds the associated area (class) names.

Presently, the OODB schema created by the Schema Extractor is specified in the DDL of the ONTOS system. Therefore, OOVR will be an ONTOS database. In future work, we will update the Schema Extractor such that it will build a schema specification in the portable Object Definition Language (ODL) proposed by the Object Database Management Group (ODMG) [10]. This would make our software independent of the back-end OODB system, which then could be any system that is ODMG-compliant.

The Program Generator, as its name suggests, generates two architectural modules (as marked by dashed arrows in Figure 2.12): the Concept Creator and the Property Loader (drawn as boxes with depth). As we see, to do its work, the Program Generator requires the OODB schema produced by the Schema Extractor. Before describing why the Program Generator module is needed, let us discuss the details of the two modules that it generates.

The Concept Creator and the Property Loader together populate the OOVR. The Concept Creator first instantiates all concepts. That is, it creates one object in the OOVR for each concept in the source CV. The class of each object is determined by the Concept/Area File, which contains the concept-to-area mapping that we have described in the previous section. Note that the OOVR database contains all its concepts when the Concept Creator finishes, but none of those concepts has any property values (as indicated by the "phantom" OOVR having the dashed cylinder in the picture). This situation is rectified by the Property Loader which provides the concepts with the values of all their attributes and relationships. It obtains these from the CSF that comes directly from the Preprocessor stage. Let us note that the Concept Creator and the Property Loader include the OOVR Schema generated by the Program Generator. In Figure 2.12, we indicate this by an arrow with label "#include."

The reason that the process of populating the OOVR is divided into two steps is because relationships are concept-to-concept (or object-to-object) references. In order to establish a relationship at a given object, the referenced objects must already exist. However, this may not be the case while the Concept Creator is carrying out its task. Therefore, it is necessary to defer the establishment of relationships until after all concepts have been created. So, in our architecture, Concept Creator first creates all the objects, and then Property Loader connects them via the appropriate relationships (and assigns their attribute values, as well).

The need for the Program Generator is dictated by the differences in structure that one finds among CVs. While we have set forth general vocabulary characteristics in Section 1.2, different CVs will certainly exhibit diverse properties and property introduction patterns. In other words, different CVs will have different OODB schemas! Both the Concept Creator and the Property Loader utilize the class definitions contained in the schema to perform their functions in the task of populating the OOVR. The distinctions in the class definitions (e.g., the diversity of class names, numbers of properties, and so on) from CV to CV require that the declaration sections of both these modules be created anew for each source CV. Fortunately, the modules' overall forms have been captured as templates, and the process of generating them is automated. As mentioned above, this leaves only the Preprocessor open to changes for different CVs.

CHAPTER 3

MODELING VOCABULARIES INTO STRUCTURALLY AND SEMANTICALLY UNIFORM CONCEPT GROUPS

In this chapter, we first prove some formal characteristics of the OOVR representation. We then describe an algorithm for partitioning a CV into its areas. Two navigation examples are then presented in Section 3.2. After that, we explain the problems we encountered during browsing at multi-rooted intersection areas. Next, we present the solution we developed to further partition multi-rooted intersection areas. The result of this process is referred to as *the singly-rooted representation.* We also state formal characteristics of the singly-rooted representation as theorems, and present an algorithm for partitioning a multi-rooted intersection area. In the last section, we apply the revised technique to the MED to produce the singly-rooted MED OOVR.

3.1 Characteristics of Areas and the Structural OOVR Schema

The methodology presented in Chapter 2 organizes concepts into areas with the same properties. Instances of each area class are structurally uniform. The areas and the OOVR schema have several characteristics which can be stated formally. We proceed now to prove several theorems regarding areas and the OOVR schema.

Lemma 1: A property-introducing concept is a root of its area. □

Proof: Let **v** be a property introducing concept introducing property p. If **v** is the concept **Entity**, then clearly **v** is a root of its area. Otherwise, **v** has parents in the vocabulary. Thus, the parent concepts of **v** in the semantic network do not have the property p and the property sets of the parents of **v** must all be different from P(**v**). Hence, none of **v**'s parents reside in **v**'s area. ■

Lemma 2: A root of a property-introducing area is a property introducing concept. □

Proof: Assume to the contrary that a root $\mathbf{r}$ of a property-introducing area A is not a property introducing concept. Let $\mathbf{v}$ be a property introducing concept contained in A. Let p be a property introduced at the concept $\mathbf{v}$. By Lemma 1, $\mathbf{v}$ is also a root of A. Since $\mathbf{r}$ is a root of A, it is not a descendant of $\mathbf{v}$, and thus it does not have the property p but then $\mathbf{r}$ and $\mathbf{v}$ must be in different areas. ■

Lemma 3: All areas have at least one root. □

Proof: Since a CV is a DAG without cycles, areas contain induced subgraphs[1] [20] of the CV. Each subgraph is also a DAG since it contains no cycle. The roots of those subgraphs will be the roots of the area. ■

Lemma 4: A property-introducing area has exactly one root. □

Proof: By Lemma 3, an area must have at least one root. Suppose to the contrary that a property-introducing area A has at least two roots $\mathbf{r}_1$ and $\mathbf{r}_2$. By Lemma 2, the root $\mathbf{r}_1$ ($\mathbf{r}_2$) is a property introducing concept introducing a property p_1 (p_2). By the uniqueness of property introduction rule, $p_1 \neq p_2$. However, $\mathbf{r}_2$ is not a descendent of root $\mathbf{r}_1$ since $\mathbf{r}_2$ is also a root of A. Hence, $\mathbf{r}_2$ does not inherit the property p_1 which is introduced only at the concept $\mathbf{r}_1$. Therefore, $\mathrm{P}(\mathbf{r}_1) \neq \mathrm{P}(\mathbf{r}_2)$ and $\mathbf{r}_1$ and $\mathbf{r}_2$ do not reside in the same area. ■

Lemmas 1, 2, and 4 together give as the following:

Theorem 1: There is a one-to-one correspondence between the property-introducing concepts, property-introducing areas, and the roots of these areas. □

[1] A graph $\mathbf{U}$ is said to be an induced subgraph of $\mathbf{V}$ if concepts $\mathbf{x}$ and $\mathbf{y}$ in $\mathbf{U}$ then the IS-A relationship between $\mathbf{x}$ and $\mathbf{y}$ is also in $\mathbf{U}$.

Corollary 1: The number of property introducing areas is equal to the number of property introducing concepts. □

Corollary 2: The number of property introducing areas is bounded by the number of different properties defined for the vocabulary. □

Proof: By Corollary 1 and the uniqueness of property introduction rule, there is at most one property introducing area for each property. Note that when several properties are introduced at the same concept, there is only one corresponding area introducing them. In such a case, the number of property-introducing areas will be smaller than the number of different properties in the vocabulary. ■

Remark: An intersection area can have multiple roots. Figure 3.1 shows such an example. Concepts in one box have the same set of properties. In this example, intersection area *D_Area* has two roots.

Lemma 5: There are only property-introducing and intersection areas. That is, every concept of the vocabulary either belongs to a property-introducing area or an intersection area. □

Proof: Let A be an arbitrary area rooted at $\mathbf{r}_A$. Let $\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_n$ $(n \geq 1)$ be the parents of $\mathbf{r}_A$. Note that $\forall i\colon 1 \leq i \leq n$, $\mathrm{P}(\mathbf{w}_i) \neq \mathrm{P}(\mathbf{r}_A)$. If the union of the property sets of $\mathbf{r}_A$'s parents is different from $\mathbf{r}_A$'s property set [i.e., $\bigcup_{i=1}^{n} \mathrm{P}(\mathbf{w}_i) \neq \mathrm{P}(\mathbf{r}_A)$], then there is some property newly introduced at $\mathbf{r}_A$. In that case, $\mathbf{r}_A$ is a property-introducing concept and A is a property-introducing area. Otherwise, the union of the property sets of $\mathbf{r}_A$'s parents is the same as $\mathbf{r}_A$'s property set [i.e., $\bigcup_{i=1}^{n} \mathrm{P}(\mathbf{w}_i) = \mathrm{P}(\mathbf{r}_A)$]. Then, by Definition 6, $\mathbf{r}_A$ is an intersection concept since $\forall i : 1 \leq i \leq n$, $\mathrm{P}(\mathrm{w}_i) \neq \mathrm{P}(\mathbf{r}_A)$, and A is an intersection area. ■

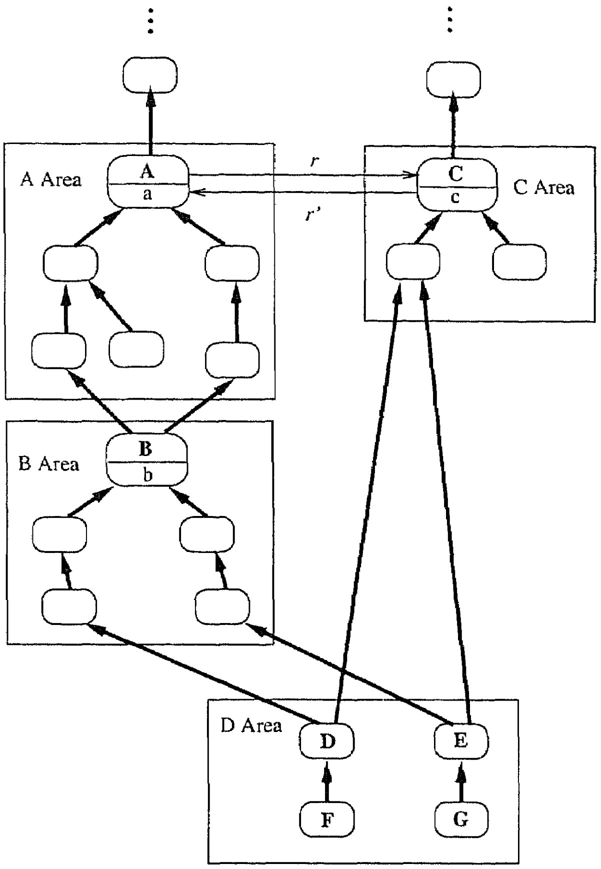


Figure 3.1 Four areas of a CV.

Theorem 2: A vocabulary is partitioned into disjoint areas which are either property-introducing areas or intersection areas. □

Proof: Areas are disjoint by definition. By Lemma 5, every area is either a property-introducing area or an intersection area. ■

Below, we show the algorithm to partition a CV into its respective areas, which is supported by the theoretical framework presented. This algorithm is designed in a top-down manner to process concepts in a CV. It takes a CV as an input and returns a set of areas. An area will be named after a concept which is a property-introducing concept or a first encountered intersection concept. We refer to these concepts as "naming concepts."

In the algorithm, $\mathbf{A_v}$ will denote a set of concepts, each of which has the same set of properties, with **v** as its naming concept. **S** is a set which holds all naming concepts. **A_Set** is a set of $\mathbf{A_v}$. At the end of the partitioning process, **A_Set** will be returned to the function's caller. Each element **v** in **S** will later be used to name an area with the format *v_Area*. Every element **v** in **S** will have an associated set $\mathbf{A_v}$ in **A_Set**. Every concept **v** has an associated counter for unprocessed parents which is denoted as "p-counter[v]." This algorithm uses two auxiliary functions: "Num_parents_of" and "Is_proper_introducing." The function Num_parents_of takes a concept as input and returns the number of its parents. Is_property_introducing is a predicate that takes a concept as input and returns "true" if it is a property-introducing concept and "false" otherwise. Note that statements with the same indentation are considered to be in the same block. A concept can be processed only if its parents have been processed. Therefore, initially the root of a CV is ready to be processed.

```
set_of_areas FUNCTION Group_Concept (CV V):

BEGIN
    // initialization
    Q := newqueue();              // Q contains concepts ready to be processed.
    A_Set := newset();            // A_Set will hold all areas.
    S := newset();                // S will hold property-introducing concepts
                                  // and "first identified" intersection concepts.
    FOR EACH concept v in V DO  // Initialize p-counter for all concepts.
            p-counter[v] := Num_parents_of(v);
    enqueue(Q, root(V));                 // Insert Entity into Q.

    WHILE ( NOT emptyqueue(Q) ) DO
            v := dequeue(Q);
            // If v introduces new properties, we generate a new introducing area.
            IF ( Is_property_introducing(v) ) THEN
                    insert(S, v);
                    $A_v$ := newset();      // Create a set $A_v$ for concepts in this area.
                    insert($A_v$, v);       // Insert v into $A_v$.
                    insert(A_Set, $A_v$);  // Insert the new set $A_v$ into A_Set.
            // If v has only one parent, v is in the same area as its parent.
```

```
        ELSE IF ( Num_parents_of(v) = 1 ) THEN
            Let $\mathbf{w}$ be the parent of $\mathbf{v}$;
            Let $\mathbf{A_u} \in$ A_Set be the set s.t. $\mathbf{w} \in \mathbf{A_u}$;
            insert($\mathbf{A_u}$, $\mathbf{v}$);
        ELSE
            // v has multiple parents $\mathbf{w}_1$, $\mathbf{w}_2$, ..., $\mathbf{w}_n$ ($n > 1$).
            IF ( for some $i$, P(v) = P($\mathbf{w}_i$)) THEN    // see Note 1
                Let $\mathbf{A_u} \in$ A_Set be the set s.t. $\mathbf{w}_i \in \mathbf{A_u}$;
                insert($\mathbf{A_u}$, $\mathbf{v}$);
            ELSE
                //Concept v is an intersection concept.
                Let flag *found* := false;              // see Note 2
                // Check whether v is the first encountered intersection
                // concept in its area.
                FOR EACH element $\mathbf{c} \in \mathbf{S}$ DO
                    IF ( NOT *found* AND P(v) = P(c) ) THEN
                    // Concept v is not the first encountered
                    // intersection concept.
                        insert($\mathbf{A_c}$, $\mathbf{v}$);
                        set flag *found* := true;
                IF ( NOT *found* ) THEN
                    // Concept v is the first encountered
                    // intersection concept.
                    insert(S, v);
                    $\mathbf{A_v}$ := newset();
                    insert($\mathbf{A_v}$, $\mathbf{v}$);
                    insert(A_Set, $\mathbf{A_v}$);
        // After we process concept v, we need to decrease the p-counter of v's
        // children by one. After the decrease, if any p-counter is equal to zero,
        // we put the associated concept into the queue since it is ready to be
        // processed.
        FOR EACH child k of v DO
            p-counter[k]−−;
            IF ( p-counter[k] = 0 ) THEN
                enqueue(Q, k);
    RETURN A_Set;
END □
```

Note 1: Formally, we can state this condition as: $\exists i$: $1 \leq i \leq n$ such that P(**v**) = P($\mathbf{w}_i$). In this case, **v** is not an intersection concept and **v** is in the same area as $\mathbf{w}_i$. All other parents with the same set of properties as $\mathbf{w}_i$ are in the same area as $\mathbf{w}_i$.

Note 2: It is possible that this intersection area might already have been identified by a previous intersection concept visit; so it is necessary to check the elements of S to determine whether this is the first time a concept with this particular property set has been visited.

As an example, let us illustrate the construction of a set **A_Set** constituting an intersection area with multiple roots. Suppose the first concept of the intersection area *D Area* in Figure 3.1 to be processed is **D**. We create a set $\mathbf{A_D}$ with **D** as its first element. Later, we will visit the concept **E**, the other root of this area. We compare its property set to that of the concepts $\{\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}\}$ in the set **S** of naming concepts. The sets of properties of concepts **A**, **B**, and **C** do not match the property set of E, but that of **D** does match. Therefore, instead of creating a new set, representing a new area, **E** will be inserted into the existing set $\mathbf{A_D}$. When **E** is processed, the p-counter of **G** is reduced from 1 to 0, and **G** is inserted into the queue. Later on, when **G** is deleted from the queue, it has only one parent **E**, and thus is added to set $\mathbf{A_D}$.

In Section 2.3 and Section 2.4, we described how to generate the OODB (or OOVR) schema from areas. It is important to note the difference between the CV's IS-A hierarchy and the subclass hierarchy of the OODB schema, though, to be sure, the latter is derived from the former. An IS-A link between two concepts in the CV indicates that one is a subconcept (or, vice versa, a superconcept) of the other. A subclass connection between a pair of area classes in the schema denotes the fact that the set of properties exhibited by the concepts of the child area is a superset of the set of properties exhibited by the concepts in the parent area. Of course, as we have just discussed, the CV's IS-A hierarchy does appear in its entirety at the instance-level of the OOVR represented by the relationship *IS-A* defined at *Entity_Area*. The IS-A hierarchy of a CV is by definition acyclic. The following theorem shows that the OODB schema derived from it is also acyclic and hence is valid.

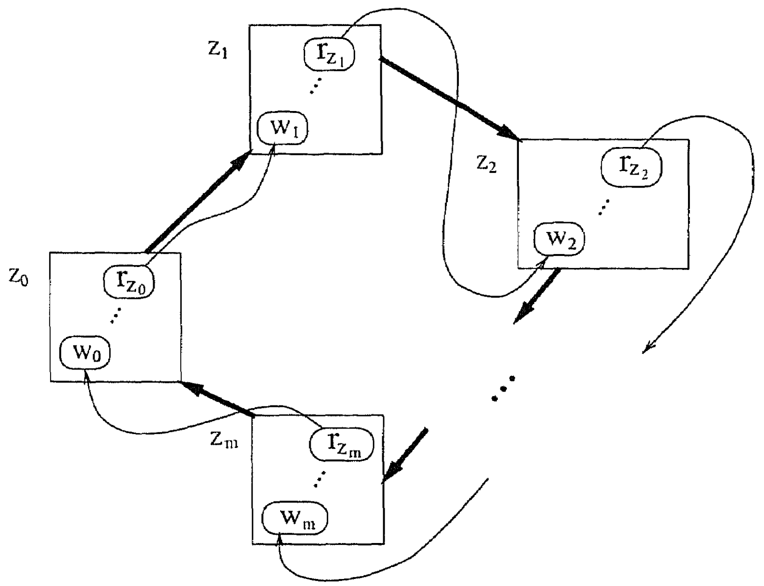


Figure 3.2 Cycle does not exist in an OOVR schema.

Theorem 3: An OOVR schema is acyclic. □

Proof: Suppose to the contrary that the OOVR schema contains a directed cycle. Each class in the schema contains more properties than any of its superclasses, either by introducing a new property (introducing class) or by inheriting properties from multiple superclasses (intersection class). Let A_1, $A_2, \ldots$, A_k be a cycle of classes. Thus the set of properties $\mathrm{P}(A_1)$ is a proper superset of the set of properties $\mathrm{P}(A_k)$. However since A_k is a subclass of A_1, the set of properties $\mathrm{P}(A_k)$ is a proper superset of the set of properties of A_1. As $\mathrm{P}(A_1)\subset\mathrm{P}(A_k)$ and $\mathrm{P}(A_k)\subset\mathrm{P}(A_1)$ cannot both hold, we found a contradiction. ■

Overall, the OOVR schema provides a structural abstraction of the underlying network of the CV. Concepts with like properties are grouped into areas which in turn are modeled as object classes; the concepts themselves become the objects of the OODB. Note that we refer to this kind of schema as a *network abstraction schema* [69].

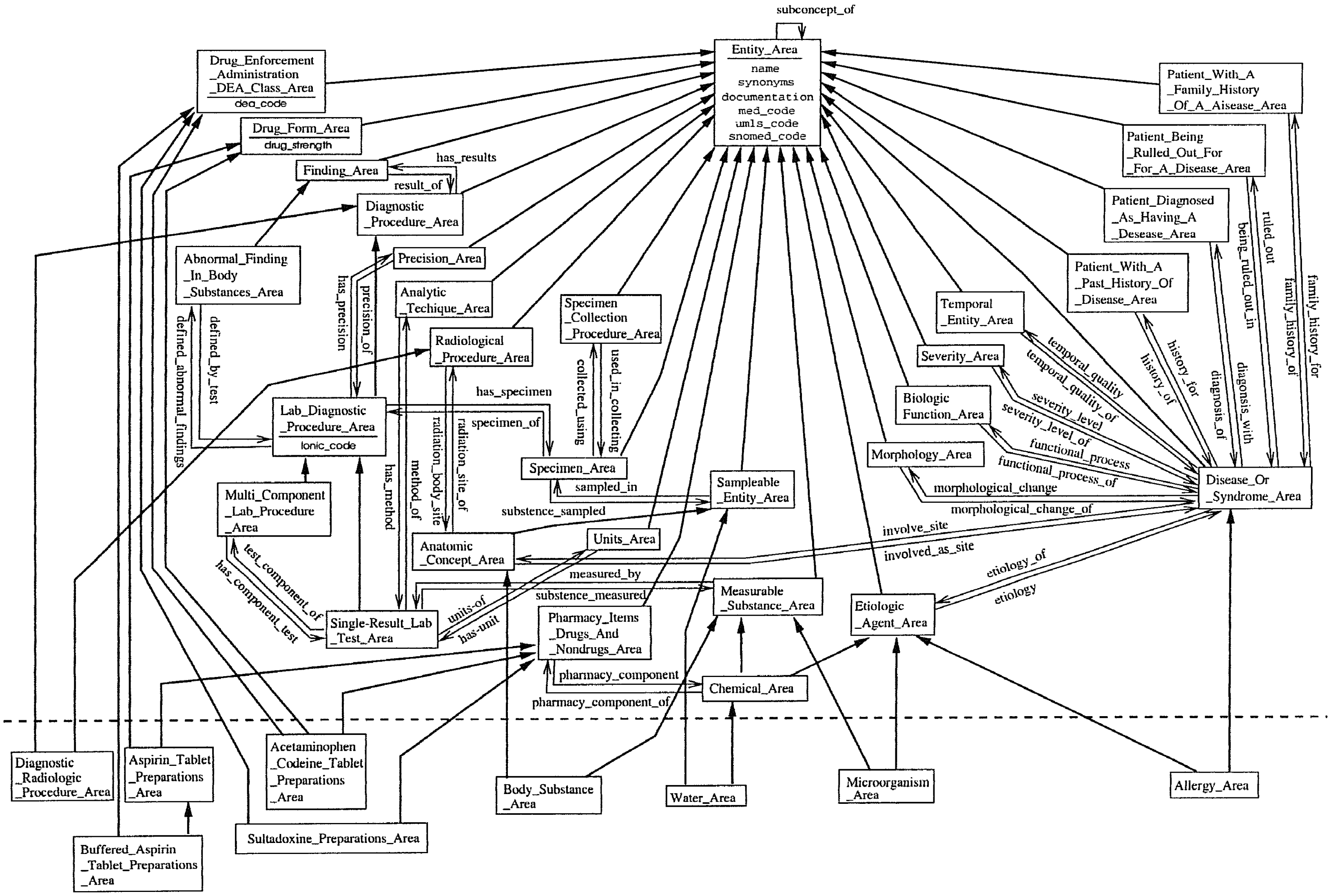


Figure 3.3 The schema of the InterMED OOVR

It is important to point out that this schema represents a substantial reduction in size from the original CV. In Figure 3.3, we show the OOVR schema of the InterMED with 2,820 concepts. Compared with the one we used in Chapter 2, this version of the InterMED has about 400 more concepts in it. Its schema contains only 39 area classes and 9 of them are intersection classes (below the dashed line). This schema is displayed on one page and can be utilized to understand the structure and content of the InterMED.

3.2 Navigation Examples

In this section, we demonstrate how the schema helps to speed up the traversal of a vocabulary.

Suppose that a user wants to search for some information in the InterMED, but does not know the name of the concept for which the information is desired. For example, suppose a user looks for a drug which treats fever and coughing for children. While the user does not remember the names of such drugs, he may recognize them when he encounters them. For this, the user needs to traverse the hierarchy of the InterMED, using his knowledge about the target to guide his choices at different levels of the hierarchy.

By combining the InterMED and the OODB schema of the InterMED into one system, we enable a combined two level traversal, which is faster than a traversal of the InterMED itself. The depth of the InterMED hierarchy is 11 which is much larger than the depth of the OODB schema which is 4. Instead of traversing the InterMED hierarchy through its many levels, we recommend a better approach. One can traverse the OODB schema until the proper class, say, *X_Area* is identified. A user will more easily be able to do this, rather than browse the vocabulary itself, as he only needs to make a very general judgment about whether the concept that he is looking for fits into the given class or not. At this point, the user needs to switch to the part of the InterMED hierarchy which contains only the concepts which belong to the class *X_Area*. The traversal runs through the levels of this subhierarchy until the desired concept is recognized (or its absence is noted).

This traversal is shorter since the number of traversing steps is bounded by the sum of the depth of the OODB schema and the subhierarchy of the InterMED belonging to the class *X_Area*. Furthermore, the traversal is also faster, since the number of subclasses of a class in the OODB schema is typically much smaller than the number of children of a concept in the InterMED. As a traversal very often requires scanning through a list of children and choosing one of them, traversal is easier at the schema level. This will also improve traversal speed. To give an intuitive analog, think about driving on a major highway to reach a target. Usually, after exiting the highway in the vicinity of the target, a person will need to travel on local streets to get to his goal. Using the schema is like driving on a highway, while traversing the InterMED hierarchy is comparable to driving on local roads.

Let us demonstrate the example traversal: looking for a drug which treats fever and coughing for children. We will list a sequence of InterMED concepts. For each concept, we list the number of children it has inside parentheses next to it. The user needs to scan this list to pick one child at every step of the traversal. We traverse through **Entity** (15), the root of the InterMED, **Pharmacy Items (drugs and non drugs)** (2), **Formulary Drug Items** (31), **Central Nervous System Agents** (8), **Analgesics And Antipyretics** (4), **Opiate Agonists** (15), **Codeine Preparations** (6), **Acetaminophen/Codeine Preparations** (2), finally leading to the target **Acetaminophen/Codeine Elixir Preparations**. The traversal of this path of 9 concepts requires the user to scan a total of 83 children.

Now we will contrast the above traversal by looking at the same problem using the OODB schema of the InterMED. We start with the root class of the schema, *Entity_Area* (23). (The number inside the parentheses is the number of subclasses of the class in the schema.) The traversal path from *Entity_-Area* is: *Pharmacy_Items_Drugs_And_Nondrugs_Area* (3) and *Acetaminophen_-Codeine_Tablet_Preparation_Area* (0). For the choices made see the schema in Figure 3.3. Since *Acetaminophen_Codeine_Tablet_Preparation_Area* is an intersection class which contains only 4 root instances, we can easily find the concept **Acetaminophen/Codeine Elixir Preparations**.

This traversal uses 3 classes with a total of 26 subclasses and 4 concepts. The total number of items to be scanned (26+4=30) is much smaller than the 83 required before.

In previous work [68], we have designed and built an interface to OOVRs that exploits their dual levels: The schema level and the concept level. Using this interface, a user can more readily traverse a CV to locate desired concepts or simply gain a general orientation.

3.3 Inadequacy of the Multi-Rooted OODB Modeling

3.3.1 Browsing Multi-rooted Intersection Areas

The traversal at the schema level is very effective when all areas are singly-rooted. In such a case, the root concept subsumes all other concepts in the area and conveys the area's general semantics. For that reason, it is reasonable to use the root at the schema level as the name of the class. However, only property-introducing areas are guaranteed to be singly-rooted.

Traversals in the context of multi-rooted intersection classes may not proceed so smoothly. This is because the name of the class is chosen arbitrarily from among the roots. Instead of conveying the general semantics for the whole area, the chosen root may capture only the essence of the concepts which are its descendants. But some concepts in the area—aside from the other roots—may not even be descendants of that root, but of other roots. Arbitrarily selecting one root tends to hide the existence of the other roots and their respective descendants. In fact, the roots may be very dissimilar from the viewpoint of semantics. Grouping them together was the result of a structural similarity. In that circumstance, it is certainly legitimate to question whether those concepts should have been grouped together in the first place.

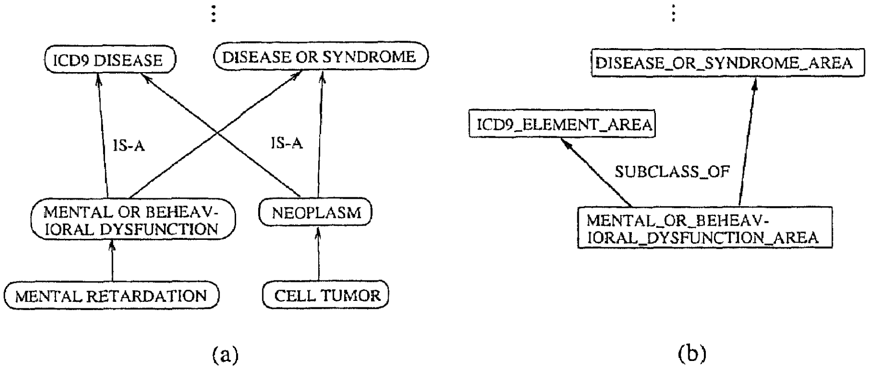


Figure 3.4 (a) Three areas including an intersection area (in the bottom box); (b) their OOVR schema

As an example, let us look at the multi-rooted intersection area shown in Figure 3.4 (a), which was gleaned from the MED. Overall, Figure 3.4 (a) contains six concepts. **ICD9 Disease** belongs to *ICD9_Element_Area* and **Disease or Syndrome** belongs to *Disease_or_Syndrome_Area* (see Figure 3.4 (b)). The concepts **Mental or Behavioral Dysfunction** and **Neoplasm** are children of both **ICD9 Disease** and **Disease or Syndrome**. They have the same structure and thus are roots of the same intersection area, *Mental_or_Behavioral_Dysfunction_Area*. Actually, in the MED, there are 29 root concepts in total for this intersection area! These include such concepts as **Infectious Disease, Disorder of Circulatory System, Disorders of Nervous System**, etc. According to Section 3.1, our mapping method randomly chooses one of them to be the naming concept. In this example, the concept **Mental or Behavioral Dysfunction** has been selected (Figure 3.4 (b)). However, one could just as easily have selected any of the other 28 concepts.

One will note that there is almost no similarity between the concepts **Mental or Behavioral Dysfunction** and **Neoplasm**, even though they have wound up in the same area. These two concepts really represent two different semantics. With **Mental or Behavioral Dysfunction** as the area's name, it is hard to imagine that **Neoplasm** also belongs there. In other words, the schema diagram does not provide a useful abstraction for assisting users in browsing the multi-rooted intersection areas of the CV.

3.3.2 Establishing Subclass Relationships for Intersection Area Classes

We have discovered an additional problem in the modeling of multi-rooted intersection areas. Our mapping method does not produce a well defined pattern for the IS-A links that traverse the boundaries of such areas. There are several equivalent modeling alternatives that properly capture the structure of the areas, but none of these is sufficient for the network interrelationships. This makes it difficult to fully comprehend these features via the OOVR schema. Figure 3.5 shows an example of this problem.

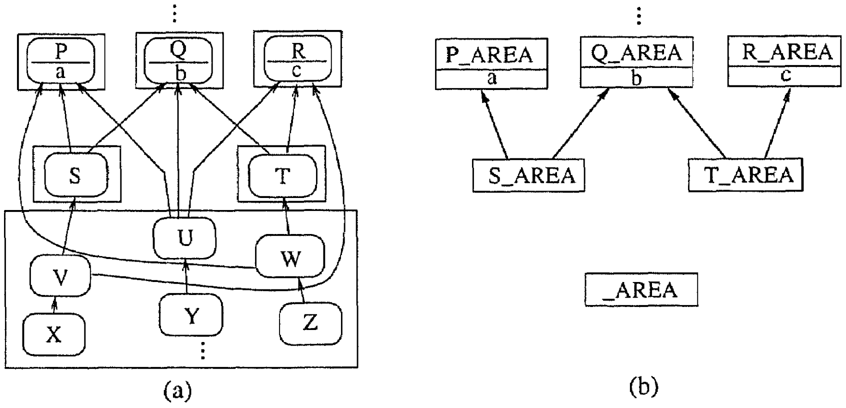


Figure 3.5 (a) CV excerpt; (b) its OOVR schema with subclass relationships from *_Area* omitted

Figure 3.5 (a) contains eleven concepts where only the top three, **P**, **Q**, and **R**, introduce new properties *a*, *b*, and *c*, respectively. Concepts **S**, **T**, and **U** have several parents which reside in different property-introducing areas and thus should belong to intersection areas. In fact, they should be roots of three different intersection areas since their property sets are different. Concepts **V** and **W** differ from concepts **S**, **T**, and **U** in that one of their parents resides in an intersection area while the other resides in a property-introducing area. However, **V** and **W** have the same property set as **U**. Our mapping method groups the concepts **U**, **V**, **W**, **X**, **Y**, and **Z** into the same intersection area. In Figure 3.5 (b), we temporarily call this intersection area *_AREA*.

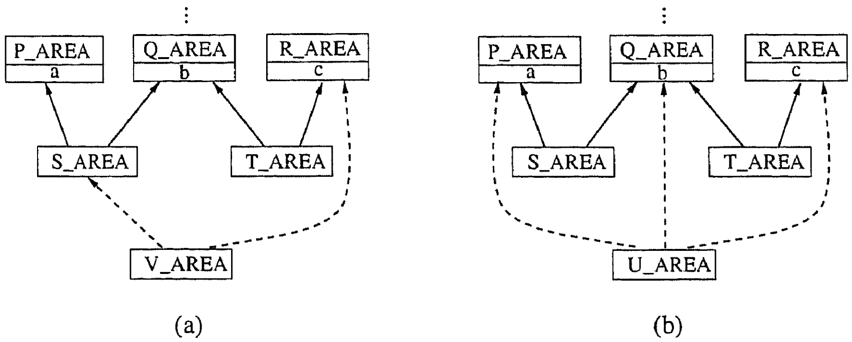


Figure 3.6 Alternative schemas for the areas in Figure 3.5 (a)

Now, there is a problem with defining correct subclass relationships for *_AREA*. (Thus, they have been omitted from Figure 3.5 (b).) Consider the concept **V**: it has concepts **S** and **R** as parents. Thus, we name that area *V_AREA* and one may define *V_AREA*'s subclass relationships to point to the classes *S_AREA* and *R_AREA* (Figure 3.6 (a)). However, the absence of a relationship between *V_AREA* and *T_AREA* may mislead the user into thinking that there is no IS-A link between any concepts of these two areas. Actually, **W** IS-A **T**. A similar problem arises if **W**'s IS-A links are used to establish *_AREA*'s subclass relationships.

Consider the concept **U**: it has parents **P**, **Q**, and **R**. Thus, we might name that area *U_AREA* to be a subclass of *P_AREA*, *Q_AREA*, and *R_AREA* (as in Figure 3.6 (b)). This schema might lead users to believe that there are no IS-A links between concepts in *U_AREA* and those in *S_AREA* or *T_AREA*. Here, though, **V** IS-A **S** and **W** IS-A **T**.

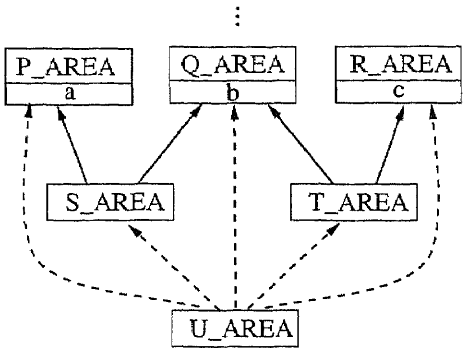


Figure 3.7 Another alternative schema for Figure 3.5 (a)

Another alternative is to define *U_AREA*'s subclass relationships to mirror all IS-A relationships of its roots. In other words, we select only one root to name the area, but use all roots to establish the subclass relationships. For each such root x, we would define subclass relationships from *U_AREA* to all area classes containing a parent of x. Using this approach, we obtain a set of parent classes for *U_AREA* which is the union of the parent classes from the alternatives considered above. In the schema of Figure 3.7, *U_AREA* has 5 parent area classes. The five subclass relationships from *U_AREA* can be misleading to users. We do not know which relationship originated from which root. Furthermore, that same schema would be generated if there existed a single root in *U_AREA* having five superconcepts in the areas *P_AREA*, . . . , *T_AREA*. Thus, this alternative is not desirable, either.

All these choices are structurally equivalent since the resulting property sets for *U_AREA* are the same. One can use any of these schemas to properly capture the properties of all the concepts. However, what has been lost is the adequacy of the OODB schema in modeling the interrelationships of the concepts in the original CV network.

3.4 Revised OODB Modeling of Intersection Areas

3.4.1 Dividing a Multi-rooted Intersection Area into Singly-rooted Partial Areas

The two problems that were presented in Section 3.3: browsing and establishing SUBCLASS_OF relationships problem, arise from placing concepts of various semantics in a single intersection area and its corresponding area class. Recall that, in general, an OODB class is a construct that gathers together objects that share the same structure (set of properties) *and* semantics. In our mapping method, most of the area classes satisfy this condition. Certainly, the structural aspect is satisfied by all area classes. Property-introduction area classes are semantically cohesive due to their unique roots, which capture the general interpretation of all their constituent concepts, and provide the areas' names. The same can be said for an intersection class having a single root. However, the synchronization of structure and semantics breaks down for multi-rooted intersection areas. All concepts of such an area have the same structure but not necessarily the same semantics because some concepts may be descendants of one root and not directly related at all to another root. It is unlikely that any single root provides appropriate "root" semantics for the entire area.

To preserve the ordinary interpretation of OODB classes as having objects with the same structure and semantics, and indeed to support effective CV access via the OOVR schema, we need to further partition a multi-rooted intersection area into separate singly-rooted partial areas. Once this is accomplished, the intersection area class can be replaced by a number of classes that have these partial areas as their respective extensions. This will ordinarily lead to the situation where several classes in the schema have the same structure, but that is not forbidden by the OODB paradigm. In the following subsection, we present a technique for carrying out this additional partitioning task.

3.4.2 Partial Areas of an Intersection Area

It is natural to place roots of a multi-rooted intersection area into different partial areas, since each root represents a distinct semantic. However, concepts may be descendants of more than one root. In such a case, they also represent distinct semantics. Thus, we create new partial areas for these kinds of concepts. Note that these newly created partial areas are considered distinct semantics groups as well. Therefore, concepts which are descendants of the roots of more than one distinct semantic group are also considered as defining new semantics in a recursive process.

In order to describe the partial areas into which a multi-rooted intersection area is partitioned, we will need some new definitions. Before getting to these, let us introduce some preliminary terminology. We will be using the term "path" to exclusively denote an *upward* path of IS-A links from some concept in the CV to one of its ancestors. The predicate "Desc" will be employed to indicate a descendant/ancestor relationship between a pair of concepts. That is, $\text{Desc}(\mathbf{x}, \mathbf{y})$ means that $\mathbf{x}$ is a descendant of $\mathbf{y}$, or, in other words, there exists a path from $\mathbf{x}$ to $\mathbf{y}$. Two concepts $\mathbf{x}$ and $\mathbf{y}$ are called *independent* if $\mathbf{x} \neq \mathbf{y}$, $\neg\text{Desc}(\mathbf{x}, \mathbf{y})$, and $\neg\text{Desc}(\mathbf{y}, \mathbf{x})$. In the following, the scope of the discussion is a multi-rooted intersection area I. The IS-A links pointing out of I are utilized only at the stage of defining the schema's SUBCLASS_OF relationships.

Definition 10: (Articulation Concept): An articulation concept (of intersection area I) is either:

1. (Base case) An intersection concept; or

2. (Recurrence) A concept $\mathbf{w}$ for which there are two independent articulation concepts $\mathbf{x}$ and $\mathbf{y}$ such that (a) $\text{Desc}(\mathbf{w}, \mathbf{x})$, and $\text{Desc}(\mathbf{w}, \mathbf{y})$, and (b) no path from $\mathbf{w}$ to $\mathbf{x}$ and no path from $\mathbf{w}$ to $\mathbf{y}$ contains another articulation concept. □

The role of the articulation concepts in a multi-rooted intersection area is corresponding to the role of the naming concepts in the vocabulary (see Function **Group_Concept** in Section 3.1). They are the concepts chosen to be roots of and name partial areas as the naming concepts were chosen to be roots of and name areas in a CV.

Definition 11: (Direct Articulation Descendant [DARD]): Let $\mathbf{v}$ and $\mathbf{w}$ be articulation concepts (in I) such that $\mathrm{Desc}(\mathbf{w}, \mathbf{v})$. The concept $\mathbf{w}$ is called a direct articulation descendant (DARD) of $\mathbf{v}$ if there exists a path from $\mathbf{w}$ to $\mathbf{v}$ that does not contain another articulation concept. □

With the definitions of articulation concept and DARD now in place, we can define the partial areas into which a multi-rooted intersection area is partitioned. Each intersection area will be divided into several partial areas (or *p-areas*, for short).

Definition 12: (P-area): A p-area (within an intersection area I) is a set of concepts containing an articulation concept $\mathbf{v}$ and all of $\mathbf{v}$'s descendants (within I) excluding its DARDs and their respective descendants. □

Again, it is important to note that a p-area will contain a single articulation concept which will be the p-area's one and only root. Any descendants that are also articulation concepts will define new p-areas. As with the areas of the CV overall, the root concept is used as the name of the p-area.

Let us now demonstrate the above formalism in the partitioning of two example multi-rooted intersection areas from a CV. The first one is X area shown in Figure 3.8, where the areas M and N are also shown. Note that X area has three roots. Its p-areas appear in Figure 3.9.

If there is no overlap among the descendants of the roots (intersection concepts) of a multi-rooted intersection area, then the only articulation concepts are the roots

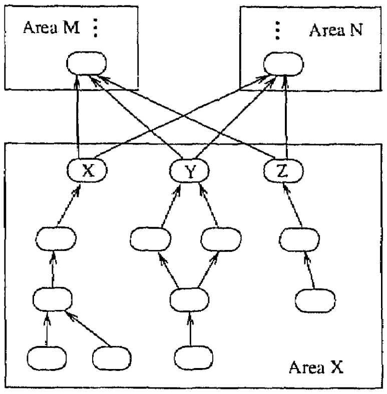


Figure 3.8 Intersection area with three roots

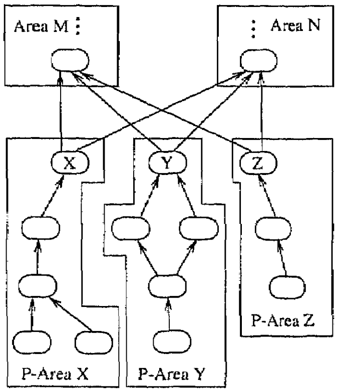


Figure 3.9 The intersection area's p-areas

themselves. This is, in fact, the case with the intersection area of Figure 3.8. In such a situation, every concept within the area is neatly grouped together with the unique root that is its ancestor. As a result, these groups form the p-areas of the original multi-rooted intersection area. Every p-area is singly-rooted.

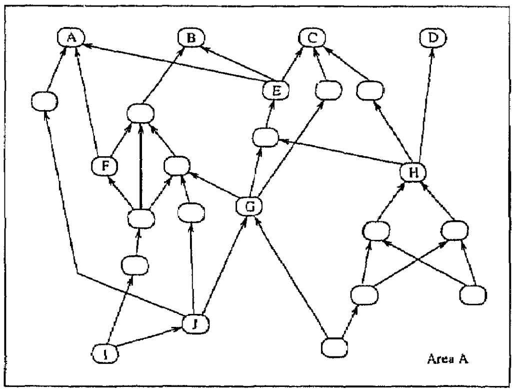


Figure 3.10 Multi-rooted intersection area with descendant overlap

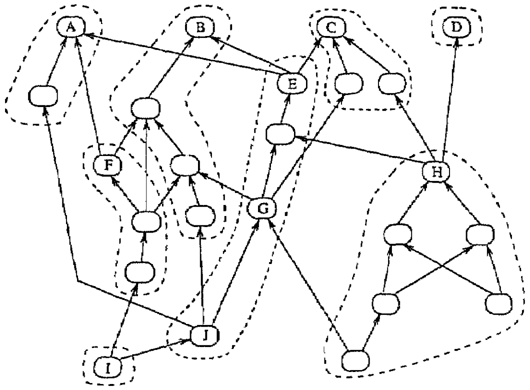


Figure 3.11 Partial areas for intersection area in Figure 3.10

The partitioning becomes more complex when the descendants of the intersection concepts overlap and create additional articulation concepts. That case is demonstrated by the intersection area *A* shown in Figure 3.10. This area has four roots: **A**, **B**, **C**, and **D**. By Definition 10, these are articulation concepts. The concepts **E**, **F**, **H**, and **I** are also articulation concepts. The eight p-areas for this intersection area are given in Figure 3.11 by dashed links.

It is interesting to note that the concept **G** is not an articulation concept even though it has paths without articulation concepts to the independent articulation concepts **B** and **C**. However, the articulation concept **E** lies on a path from **G** to B (and also on a path from **G** to **C**). Thus, **G** is not an articulation concept and, in fact, belongs to the p-area rooted at **E**. This relies on 2.(b) of Definition 10. For the same reason, the concept **J** is not an articulation concept. It, too, belongs to **E**'s p-area.

In the following, we prove that the p-areas partition the multi-rooted intersection area.

Lemma 6: Let a non-articulation concept **v** have multiple articulation ancestor concepts $\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n$, $(n > 1)$. For any i and j such that $i \neq j$, if $\mathbf{z}_j$ is a descendant of $\mathbf{z}_i$, then **v** does not belong to the p-area rooted at $\mathbf{z}_i$. □

Proof: If concept $\mathbf{z}_j$ is a descendant of $\mathbf{z}_i$, then there must exist a DARD $\mathbf{z}_k$ (possibly $\mathbf{z}_j$ itself) of $\mathbf{z}_i$ on some path between $\mathbf{z}_i$ and $\mathbf{z}_j$. Concept $\mathbf{z}_k$ is an ancestor of **v** since $\mathbf{z}_j$ is an ancestor of **v**. By Definition 12, **v** will be excluded from the p-area rooted at $\mathbf{z}_i$ as a descendant of the DARD $\mathbf{z}_k$. ■

Lemma 7: Every concept belongs to at least one p-area. □

Proof: If **v** is an articulation concept, then it is the root of its own p-area.

Assume to the contrary that a non-articulation concept **v** does not belong to a p-area. Since **v** is in an intersection area, it must have one or more articulation ancestors $\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_n$ $(n \geq 1)$ which are roots of p-areas $P_1, P_2, \ldots, P_n$, respectively. By assumption, **v** does not belong to any of the P_i's. By Definition 12, **v** is excluded from P_i ($1 \leq i \leq n$) because there exists a DARD of $\mathbf{r}_i$ (call it $\mathbf{r}_j$, $i \neq j$) such that Desc($\mathbf{v}$, $\mathbf{r}_j$). This implies Desc($\mathbf{r}_j$, $\mathbf{r}_i$). Similarly, we see that there exists an articulation concept $\mathbf{r}_k$ to exclude $\mathbf{v}$ from the p-area P_j. Repeating this n times, we can form a sequence with $n + 1$ articulation concepts starting at $\mathbf{r}_i$. Each concept in this sequence is a descendant of its predecessor. However, by assumption, $\mathbf{v}$ has only n articulation ancestors. Therefore, some concept must appear more than once in the sequence. This implies there is a cycle in the IS-A hierarchy of the CV—a contradiction. ■

Lemma 8: P-areas are disjoint. □

Proof: If $\mathbf{v}$ is an articulation concept, then by Definition 12, it is the root of a p-area of its own.

Assume to the contrary that a non-articulation concept $\mathbf{v}$ belongs to two p-areas X and Y, rooted at $\mathbf{r}_X$ and $\mathbf{r}_Y$, respectively. Since $\mathbf{v}$ belongs to the p-area X, then by Definition 12 there is no path from $\mathbf{v}$ to $\mathbf{r}_X$ which contains other articulation concepts. Similarly, since $\mathbf{v}$ belongs to the p-area Y, there is no path from $\mathbf{v}$ to $\mathbf{r}_Y$ which contains other articulation concepts. Note that the roots $\mathbf{r}_X$ and $\mathbf{r}_Y$ must be independant concepts. Otherwise, if $\mathbf{r}_X$ is a descendant of $\mathbf{r}_Y$, then by Lemma 6, $\mathbf{v}$ does not belong to p-area Y rooted at $\mathbf{r}_Y$. Similarly, $\mathbf{r}_Y$ cannot be a descendant of $\mathbf{r}_X$. But by Definition 10, if concept $\mathbf{v}$ has two independent articulation concepts ancestors $\mathbf{r}_X$ and $\mathbf{r}_Y$ such that no path from $\mathbf{v}$ to $\mathbf{r}_X$ or $\mathbf{r}_Y$ contains another articulation concept, then $\mathbf{v}$ itself is an articulation concept—a contradiction. ■

Lemmas 7 and 8 together give us:

Theorem 4: The p-areas of a multi-rooted intersection area partition the area. □

In the following, we present the algorithm that groups concepts in a multi-rooted intersection area I into p-areas. $\mathbf{A_v}$ is a set of concepts within one p-area rooted at **v**. **PA_Set** is a set of $\mathbf{A_v}$'s. At the end of the partitioning process, **PA_Set** will be returned to the caller. Every concept **v** has an associated counter for "unprocessed parents" which is denoted as "p-counter[v]." This algorithm uses one auxiliary function "Num_parents_of" which takes a concept **v** and a multi-rooted intersection area I as input, and returns the number of **v**'s parents in I. **Art**[**v**] contains the root of the p-area that **v** belongs to. Statements with the same indentation are in the same block. Unlike the algorithm introduced in Section 3.1, all concepts within a multi-rooted intersection area I have the same set of properties. Therefore, we cannot use the previous algorithm to find p-areas in I. Similar to the algorithm introduced in Section 3.1, we use a top-down process to find p-areas in I. A concept can be processed only if its parents have been processed. Therefore, initially all roots of a multi-rooted intersection area are ready to be processed.

```
set_of_p-areas FUNCTION P-AREA_Partition(area I):
BEGIN
    // initialization
    Q := newqueue();              // Q contains concepts ready to be processed.
    PA_Set := newset();                 // PA_Set will hold all p-areas.
    FOR EACH concept v in I DO    // Initialize p-counter and art[v].
            p-counter[v] := Num_parents_of(v, I);
            art[v] := ' ';
    FOR EACH root v of I DO             // Put roots of I into the queue since
            enqueue(Q, v);              // they are ready to be processed.

    WHILE (NOT emptyqueue(Q)) DO
            v := dequeue(Q);
            R_Set := newset();          //R_Set will hold the arts of v's parents.
            FOR EACH parent c of v in I DO
                    insert(R_Set, art[c]);
            IF ( |R_Set| = 1 ) THEN
                    // All parents of v are in one area.
                    // Concept v will be in the same p-area as its parents.
                    // Concept v's parents are in the p-area rooted at u.
                    Let u be the (only) element in R_Set;
                    insert($A_u$, v);             // Insert v into set $A_u$.
                    art[v] := u;                  // Assign art[v] to be the articulation
                                                  // concept u.
```

```
        ELSE IF ( $|\mathbf{R\_Set}| > 1$ AND $\exists$ u in R_Set which is a descendant
                of all other nodes in R_Set ) THEN
                // see Note 1
                insert($\mathbf{A_u}$, v);          // Insert v into set $\mathbf{A_u}$.
                art[v] := u;                // Assign art[v] to be the articulation
                                            // concept u.
        ELSE
                // Concept v is an articulation concept.
                // In case $|\mathbf{R\_Set}| = 0$, v is an intersection concept (root of $I$).
                $\mathbf{A_v}$ := newset();         // Create a new set $\mathbf{A_v}$ to hold
                                            // concepts in this p-area.
                insert($\mathbf{A_v}$, v);          // Insert articulation concept v
                                            // into set $\mathbf{A_v}$.
                insert(PA_Set, $\mathbf{A_v}$);     // Insert $\mathbf{A_v}$ into PA_Set.
                art[v] := v;                // Assign art[v] to be itself since v is
                                            // an articulation concept.
        // After we process concept v, we need to decrease the p-counter of v's
        // children by one. After the decrease, if any p-counter is equal to zero,
        // we put the associated concept into the queue since it is ready to be
        // processed.
        FOR EACH child k of v in $I$ DO
                p-counter[k]−−;
                IF ( p-counter[k] = 0 ) THEN
                        enqueue(Q, k);
    RETURN PA_Set;
END □
```

Note 1: By Theorem 4, **v** must belong to one p-area. In this case, **v** is not an articulation concept. Concept **v** will belong to a p-areas, say *Q_PArea*, rooted at **u** which is a descendant of all other elements in **R_Set**. In this case we can always find such an element **u** in **R_Set** since **R_Set** has more than one element, there are no two independant concepts, and a CV is acyclic.

Let us use Figure 3.11 to illustrate the partitioning process of a certain intersection area. Assume the roots of this area were inserted into the queue in the following order: **A**, **B**, **C**, and **D**. When we process concept **M** (dequeue it from **Q**), we will generate a set **R_Set** with one element **art**[B] (equal to **B**). Since there is only one element in **R_Set**, **M** will be inserted into $\mathbf{A_B}$ and **art**[M] will be assigned

B. Next, we are decreasing the p-counters of M's children by one. We insert **F** and N into the queue since their p-counters are zero. When we process concept **F**, the **R_Set** of **F** will have two elements, **A** and B, which are **art**[**A**] and **art**[M], respectively. Since **A** and **B** are independent, **F** is an articulation concept. It will be inserted into a new set $\mathbf{A_F}$. The variable **art**[**F**] will be assigned **F**. For processing concept **G**, the algorithm will generate **R_Set** with three elements: **art**[N] = B, **art**[K] = E, and **art**[L] = **C**. Since E is a descendant of both **B** and **C**, **G** will not be considered an articulation concept. Instead, **G** will be inserted into the p-area rooted at **E**. In other words, **G** will be inserted into $\mathbf{A_E}$, and **art**[G] will be assigned **E**.

3.4.3 The Revised OOVR Schema

After a multi-rooted intersection area is partitioned into its respective p-areas, the portion of the OOVR schema that captures it can be defined. Instead of defining a single area class to represent the entire intersection area, a separate class is defined for each of its p-areas. Such a class is called a *p-area class*. As with the original intersection class, the p-area classes do not intrinsically define any properties of their own; all are obtained via inheritance. Again, it is important to note that the p-area classes are intended as concept representations that promote better dissemination of the semantics of the underlying CV. Their purpose is not to capture structural aspects, which in fact was done properly by the original monolithic intersection class. The growth of the size of the schema is not a concern in the inclusion of the p-area classes.

The subclass relationships of a p-area class are defined with respect to the parentage of the (unique) root in a manner analogous to that for an area class. The schemas for the intersection areas of Figure 3.8 and Figure 3.10 are shown, respectively, in Figure 3.12 and Figure 3.13. Note that the name of a p-area class is

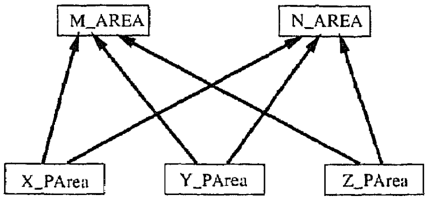


Figure 3.12 Schema for intersection area in Figure 3.8

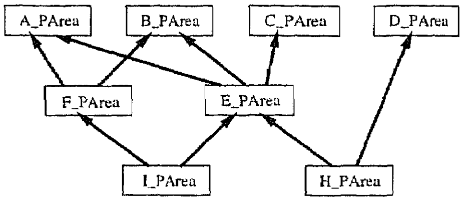
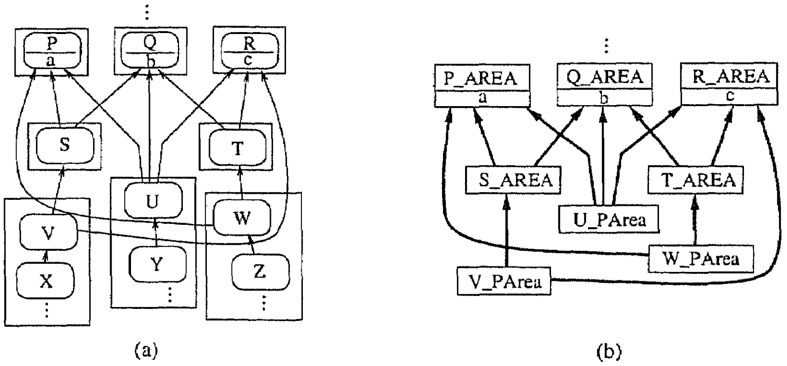


Figure 3.13 Schema for intersection area in Figure 3.10

created by adding the suffix "PArea" to the end of the root's name. In Figure 3.12, we see three p-area classes (along with two area classes) and six subclass relationships. Figure 3.13 has eight p-area classes and nine subclass relationships.

Note that shortcuts are omitted among p-area classes. Let P be a p-area rooted at $\mathbf{r}_P$ have n parents, $P_1, P_2, \ldots, P_n$ $(n \geq 1)$ rooted at $\mathbf{r}_{P_1}$, $\mathbf{r}_{P_2}, \ldots,$ $\mathbf{r}_{P_n}$, respectively. If $\mathbf{r}_{P_i}$ is an ancestor of $\mathbf{r}_{P_j}$ $(i \neq j)$, then the SUBCLASS_OF relationship from P to P_i will be a shortcut. This kind of link will be omitted from the OOVR schema.



(a)

(b)

Figure 3.14 (a) P-areas for Figure 3.5 (a); (b) the classes for those p-areas

Overall, the partitioning of multi-rooted intersection areas into p-areas that comprise singly-rooted DAG structures provides a better abstraction level for browsing and searching the CV. In addition, this enhanced partitioning of the CV solves the problem of establishing informative subclass relationships in the OOVR schema. That is, these subclass relationships more properly reflect the IS-A relationships which cross p-areas of the underlying CV. In Figure 3.14 (a), we show the p-areas for the CV excerpt which appeared in Figure 3.5 (a). Its schema appears in Figure 3.14 (b).

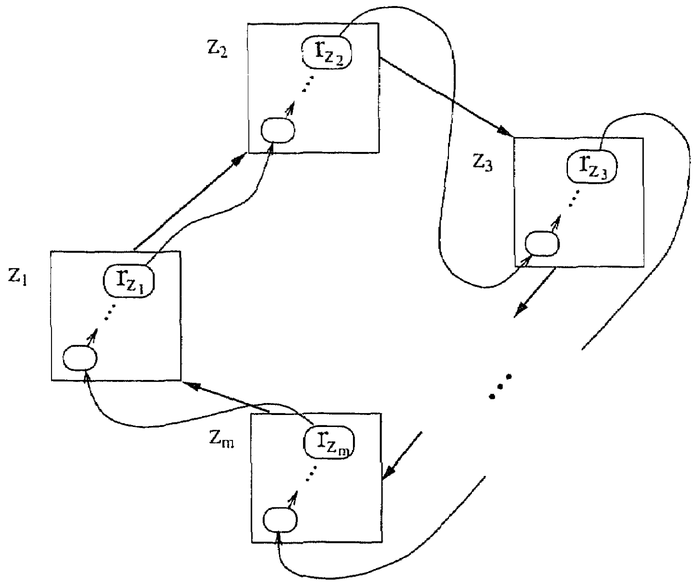


Figure 3.15 Cycle does not exist among p-area classes in the OOVR schema.

Theorem 5: There are no cycles in the singly-rooted schema containing singly rooted area classes and p-area classes. □

Proof: Suppose to the contrary that the schema contains a cycle of SUBCLASS_OF relationships. Classes represent p-areas or areas. Both kinds are singly rooted classes since multi-rooted intersection classes were replaced by singly rooted p-area classes. Assume the cycle has $m+1$ classes Z_0, Z_1, ..., Z_m such that Z_i is a SUBCLASS_OF Z_{i+1} ($i < m$) and Z_m is a SUBCLASS_OF Z_0 (see Figure 3.15). Let the roots of Z_0, Z_1, ..., Z_m be $\mathbf{r}_{z_0}$, $\mathbf{r}_{z_1}$, ..., $\mathbf{r}_{z_m}$, respectively. According to the construction of the schema, for each class Z_i ($0 \leq i < m$), there is an IS-A connection from the root $\mathbf{r}_{z_i}$ of Z_i to a concept $\mathbf{w}_{i+1}$ in Z_{i+1}. Also, $\mathbf{r}_{z_m}$ IS-A $\mathbf{w}_0$ in Z_0. Since Desc($\mathbf{w}_{i+1}$, $\mathbf{r}_{z_{i+1}}$), then Desc($\mathbf{r}_{z_i}$, $\mathbf{r}_{z_{i+1}}$). Similarly, we see that Desc($\mathbf{r}_{z_i}$, $\mathbf{r}_{z_{i+1}}$), Desc($\mathbf{r}_{z_{i+1}}$, $\mathbf{r}_{z_{i+2}}$), ..., Desc($\mathbf{r}_{z_{i-1}}$, $\mathbf{r}_{z_i}$). But this implies that a cycle of concepts exists in the CV—a contradiction. ■

Table 3.1 Multi-rooted intersection classes in the OOVR schema for the MED

Intersection classes	Number of roots	Number of concepts
Intersection class 1	2	106
Intersection class 2	29	19364
Intersection class 3	965	6967
Intersection class 4	31	324
Intersection class 5	11	182
Intersection class 6	9	170
Intersection class 7	15	40
Intersection class 8	42	43
Intersection class 9	2	175
Intersection class 10	3	3
Intersection class 11	2	52
Intersection class 12	3	96
Intersection class 13	124	124
Intersection class 14	16	16
Total: 14	Total: 1254	Total: 27662

3.5 Applying the Revised Mapping to an Existing CV

In this section, we will apply our revised modeling approach to an existing OOVR that contains multi-rooted intersection classes [69]. The OOVR was originally obtained from the MED 1996 version [12], which contains about 48,000 concepts and 61,000 IS-A links. The OOVR's original schema consisted of 90 area classes: 53 property-introducing classes and 37 intersection classes. Of the 37 intersection classes, 14 are multi-rooted. Each of these is listed in Table 3.1 along with its number of roots and number of constituent concepts. The average number concepts of these of classes is 1,975. Note that the number of concepts belonging to such a class can be huge. For example, Class 2 in Table 3.1 contains 29 roots and 19,364 concepts.

In Table 3.1, there are nine classes where the descendants of the various roots form disjoint sets. One of them is Class 1, *BODY_SUBSTANCE_AREA*,

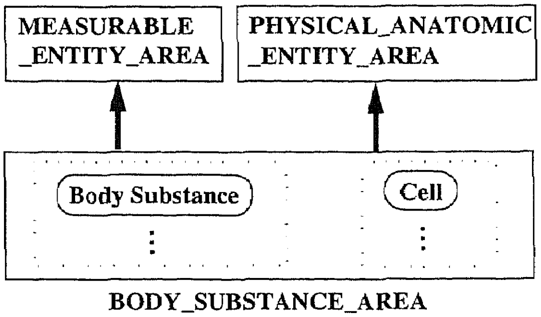


Figure 3.16 Multi-rooted intersection class with disjoint descendants of roots in original MED-OOVR schema

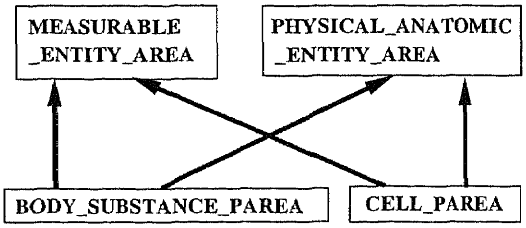


Figure 3.17 Refined schema from Figure 3.16

which has two roots, **Body Substance** and **Cell**, and a total of 106 concepts (Figure 3.16). Due to the disjointness, the only articulation points in the area are the roots. Thus, applying our revised modeling approach, we get two p-area classes *BODY_SUBSTANCE_PAREA* and *CELL_PAREA* in the new schema, which replace *BODY_SUBSTANCE_AREA* in the old schema. Both p-area classes have the superclasses *MEASURABLE_ENTITY_AREA* and *PHYSICAL_ANATOMIC_-ENTITY_AREA* (Figure 3.17). The other eight classes having disjoint descendants of their roots are 6, 7, 8, 9, 10, 11, 13, and 14.

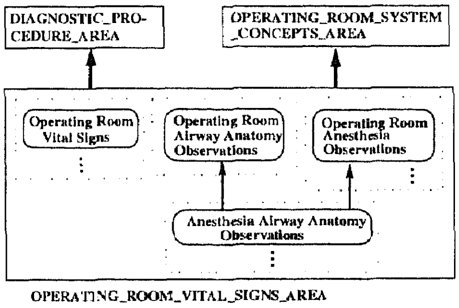


Figure 3.18 Multi-rooted intersection class with non-disjoint descendants in original MED-OOVR schema

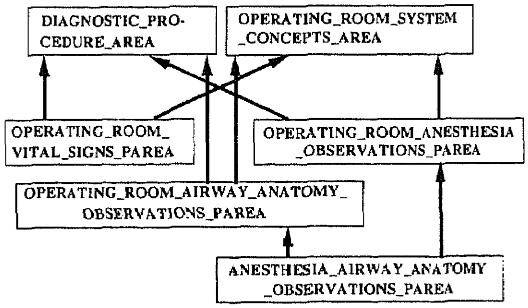


Figure 3.19 Refined schema from Figure 3.18

The remaining classes in Table 3.1 have roots whose descendants overlap. One of them is Class 12, *OPERATING_ROOM_VITAL_SIGNS_AREA*, which has three

Table 3.2 P-area classes in singly-rooted MED-OOVR schema

Intersection classes in the original schema	Number of roots	Number of p-area classes in the singly-rooted schema	Average number of concepts per class
Intersection class 1	2	2	53
Intersection class 2	29	168	115
Intersection class 3	965	1038	7
Intersection class 4	31	48	7
Intersection class 5	11	16	11
Intersection class 6	9	9	19
Intersection class 7	15	15	3
Intersection class 8	42	42	1
Intersection class 9	2	2	87
Intersection class 10	3	3	1
Intersection class 11	2	2	26
Intersection class 12	3	4	24
Intersection class 13	124	124	1
Intersection class 14	16	16	1
Total: 14	1254	1489	Average: 19

roots, **Operating Room Vital Signs**, **Operating Room Anesthesia Observations**, and **Operating Room Airway Anatomy Observations**, and a total of 96 concepts (Figure 3.18). This class has four articulation concepts, three of which are the roots. The other is **Anesthesia Airway Anatomy Observations**, a descendant of both **Operating Room Anesthesia Observations** and **Operating Room Airway Anatomy Observations** (Figure 3.18). The four p-area classes which supplant the original intersection class are shown in Figure 3.19.

Table 3.2 summarizes the results of applying the revised mapping to the MED. Previously, we had fourteen multi-rooted intersection classes with 1,254 roots in total. The new schema contains 1,489 p-area classes. Their average size is nineteen concepts. This should be compared to the average size of 1,975 concepts of the intersection classes that were replaced. This more detailed abstraction level provides a set of smaller and more manageable semantic units and facilitates better navigation of the vocabulary.

CHAPTER 4

THE REVISED OOVR PRESENTATION WITH IS-A$'$ SEMANTIC RELATIONSHIPS

In Chapter 2, a methodology was introduced to map a CV into an OODB. The mapping created what we call the OOVR system. During the process, an area diagram was generated. We call this area diagram a "original area diagram," since each area has a structure of a unique set of properties. The OOVR schema is referred to as the original OOVR schema (or OOVR schema in short). Chapter 3 presented the revised mapping methodology which generates a revised version of the OOVR, including a singly-rooted schema. This process partitions some areas into "partial areas." As we mentioned in the previous chapter, the singly-rooted schema overcomes several inadequacies in our original mapping methodology.

However, there are several more problems that it does not address. Section 4.1 describes these problems with detailed examples. In Section 4.2, we present a framework to resolve these problems. In Section 4.3, we demonstrate that this framework properly handles the examples from Section 4.1. In Section 4.4, we exhibit some interesting theoretical characteristics of the singly-rooted OOVR representation with proofs.

4.1 Problems for Utilizing the OODB Paradigm to Model a CV

In this section, examples of three different kinds of problems are presented. In order to understand these problems, we need to recall the nature of the OOVR (or singly-rooted) schema and how it is used. The OOVR schema is compact in size. It provides an abstraction of a CV. Users (including the CV designers) can employ it as an aid in performing their tasks more easily. Several browsing examples have been introduced in the earlier chapters. Note that in order to perform the browsing, the links between area classes, that is, the SUBCLASS_OF relationships, must reflect the correct IS-A configuration of the CV. Otherwise, the area diagrams will mislead the users. To be specific, we need to define a browsing path on both the concept level and the area (class) level.

Definition 13: (Concept Level Browsing Path) A concept level browsing path is a sequence of concepts $\mathrm{P}_C = (\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_n)$ such that $\mathbf{c}_{i+1}$ IS-A $\mathbf{c}_i$, $1 \leq i < n$. □

Definition 14: (Schema Level Browsing Path) A schema level browsing path is a sequence of areas (and p-areas) $\mathrm{P}_A = (A_1, A_2, \ldots, A_k)$ such that area class A_{i+1} is SUBCLASS_OF area class A_i. □

Now we can set the condition for the SUBCLASS_OF relationships to properly reflect the IS-A relationships in the CV. For every browsing path $\mathrm{P}_C = (\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_n)$ on the concept level of a CV, there should be a parallel browsing path $\mathrm{P}_A = (A_1, A_2, \ldots, A_k)$ on the schema level which satisfies the following conditions.

1. Concept $\mathbf{c}_1$ is an instance of area A_1.
2. Concept $\mathbf{c}_n$ is an instance of area A_k.
3. There exists a partition of P_C into disjoint consecutive subpaths, say $\mathbf{c}_{i_1}, \ldots, \mathbf{c}_{i_e}$, which are paths in the induced subnetwork of an area A_j, $1 \leq j \leq k$.

Figure 3.6 (a) shows such an example where users may think concepts between *T_Area* and *U_Area* do not have IS-A relationships. From this perspective, the SUBCLASS_OF relationships should capture the abstraction of the IS-A (specialization/generalization) relationships in the CV.

On the other hand, the IS-A relationships in a CV carry out another important function which is property inheritance. In the OODB paradigm, the SUBCLASS_OF relationships provide property inheritance between classes. The SUBCLASS_OF relationships between area classes ensure that concepts in an area have the correct property set. The problems that we will present in this subsection refer to conflicts between these two roles of the IS-A relationships.

4.1.1 The Removing of the "Short Cut" Relationships

Let us consider the first kind of such problems due to the lack of representation of IS-A relationships by the SUBCLASS_OF relationships. Figure 4.1 shows an excerpt CV with areas described as boxes. Note that concepts **w** and **x** have the same set of properties. These four concepts will generate three property-introducing areas as shown in Figure 4.2.

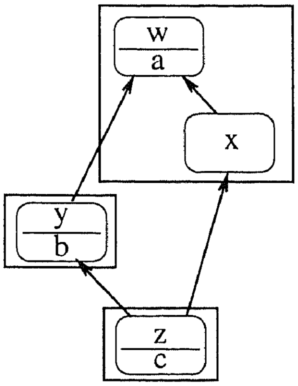


Figure 4.1 An example CV with a "short cut" problem

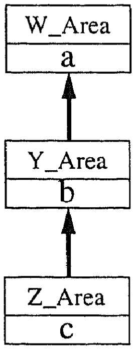


Figure 4.2 The original and singly-rooted schema for the CV shown in Figure 4.1

Note that the root **z** of the area *Z_Area* has parents in *W_Area* and *Y_Area*, where *Y_Area* is a child of *W_Area*. Thus we will need to set SUBCLASS_OF relationships from the *Z_Area* class to the *W_Area* class and *Y_Area* class. Also the *Y_Area* class is SUBCLASS_OF the *W_Area* class. In such a case, the link between the *Z_Area* class and the *W_Area* class is called a "short cut" relationship. In general, a SUBCLASS_OF relationship from class A to class B is called a short cut if there is a path of at least two SUBCLASS_OF relationships from A to B. Referring to the SUBCLASS_OF relationship using family terminology, a short cut is a parent link to an ancestor. In the OODB paradigm, it does not make any sense to put a SUBCLASS_OF relationship from *Z_Area* to *W_Area* or at any short cut, since the SUBCLASS_OF relationships are transitive by definition. Such links will be removed from the OOVR schema (see Figure 4.2). In other words, there will not be a SUBCLASS_OF relationship between area classes *Z_Area* and *W_Area*. From the view point of property inheritance, class *Z_Area* will have the correct property set even without the "short cut" link in the schema. This omission will not have any effect on property inheritance since class *Z_Area* inherits class *W_Area*'s properties via class *Y_Area*.

However, from the point of view of connection between areas, the removal of such links results in some information loss. The resulting problems can be viewed from two aspects. First, given a schema like Figure 4.2, there is no way we can tell whether there was originally a short cut link or not. There is no such information in the OOVR schema indicating whether we removed a short cut link or not. Second, a schema like Figure 4.2 has no schema level browsing path parallel to the concept level browsing path (**w**, **x**, **z**).

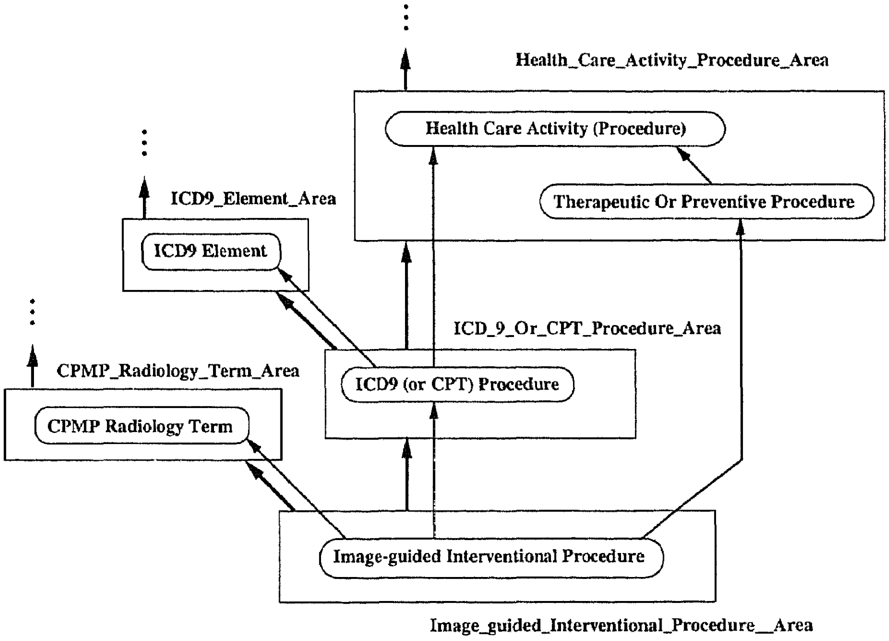

Health_Care_Activity_Procedure_Area
Health Care Activity (Procedure)
Therapeutic Or Preventive Procedure
ICD9_Element_Area
ICD9 Element
ICD_9_Or_CPT_Procedure_Area
ICD9 (or CPT) Procedure
CPMP_Radiology_Term_Area
CPMP Radiology Term
Image-guided Interventional Procedure
Image_guided_Interventional_Procedure__Area

Figure 4.3 A "short cut" example from the MED

Figure 4.3 shows such an example from the MED. In this excerpt of the MED, there are five classes: *Health_Care_Activity_Procedure_Area*, *ICD9_Element_Area*, *ICD9_Or_CPT_Procedure_Area*, *CPMC_Radiology_Term_Area*, and *Image_guided-*

_Interventional_Procedure_Area. Note that only a portion of their concepts are shown in this picture. The root of *Image_guided_Interventional_Procedure_Area*, **Image-guided Interventional Procedure**, has three parents: **ICD9 (or CPT) Procedure** and **CPMC Radiology Term** and **Therapeutic Or Preventive Procedure**. Due to the first two parents there are SUBCLASS_OF relationships to their own areas. The link to the third parent is a "short cut" link. When we browse this schema, we have no idea that concepts in *Image_guided_Interventional_Procedure_Area* have parents residing in *Health_Care_Activity_Procedure_Area* since the SUBCLASS_OF relationship between these two areas was removed as a short cut.

4.1.2 The Problem of "Missing" SUBCLASS_OF Relationships

Similar to "short cut" relationships, we have found that two unrelated area classes may have subsumption (IS-A) relationships connecting their instances in the concept level networks. This kind of inaccuracy makes the use of the OODB schema for helping in browsing the controlled vocabulary less effective. We called this kind of problem the "missing" SUBCLASS_OF relationships problem (or missing link problem, for short).

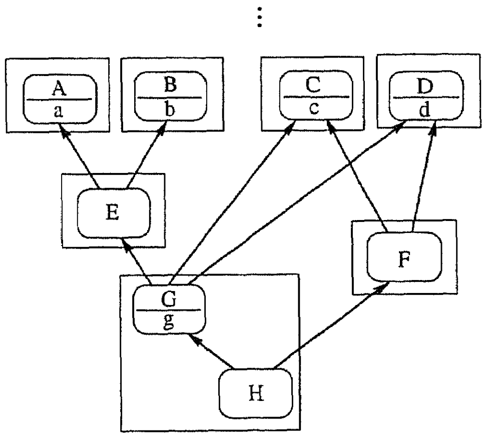


Figure 4.4 A CV with a missing link problem

An example of such a case is shown in Figure 4.4 which contains eight concepts, **A** through **H**. The concepts **A**, **B**, **C**, **D**, and **G** are property-introducing concepts and introduce attributes a, b, c, d, and g, respectively. The concepts **E** and **F** are intersection concepts with property sets $\{a, b\}$ and $\{c, d\}$, respectively. In Figure 4.4, concepts within a box have the same set of properties and constitute an area. Concepts **G** and **H** have the same property set $\{a, b, c, d, g\}$. Therefore, they should reside in one area, named *G_Area* since **G** is the root of this area. By our mapping method, the parent areas of the *G_Area* are the *C_Area*, the *D_Area*, and the *E_Area*.

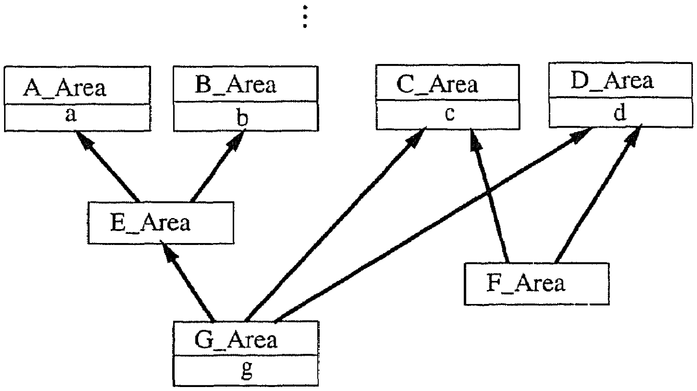


Figure 4.5 The singly-rooted schema of the CV shown in Figure 4.4

Since this example does not contain multi-rooted intersection areas, the original and singly-rooted schema are the same (Figure 4.5). The schema shown in Figure 4.5 shows no relationship between the *G_Area* class and the *F_Area* class. However, concept **H** in the *G_Area* class is a subconcept of concept **F** in the *F_Area* class, but this subconcept relationship will not be represented in the area diagram and in the schema, since concept **H** is not the root of its area. Thus there is no schema level browsing path corresponding to the concept level browsing path (**D**, **F**, **H**) in the schema Figure 4.5.

The OODB schema in Figure 4.5 is not incorrect from the perspective of capturing property inheritance, since concepts are instances of the area classes with the correct property sets. The only problem is that when we use the OODB schema as an abstract network diagram, the missing links may cause browsing difficulties. Adding a SUBCLASS_OF relationship between *G_Area* and *F_Area* will create two additional problems. First, it will create two short cut links in the schema (from *G_Area* to *C_Area* and to *D_Area*). Short cut SUBCLASS_OF relationships should not be in the schema. Second, if we define *G_Area* to be a SUBCLASS_OF *F_Area*, the schema may mislead users regarding the fact that **G** IS-A **F** does not hold.

4.1.3 The Partitioning Boundary for Multi-rooted Intersection Areas

The missing link problem may also happen on the singly-rooted schema. In our revised mapping methodology, multi-rooted intersection areas are further partitioned into singly-rooted p-areas. This partitioning process is bounded within a certain multi-rooted intersection area. Any IS-A links from an area pointing outside of this area is ignored. In other words, the result of a further partitioning of a multi-rooted intersection area will not be propagated to its descendant.

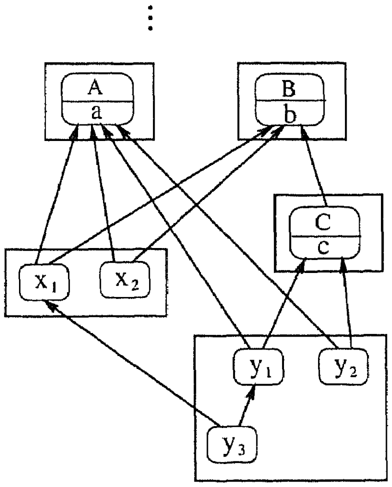


Figure 4.6 An example CV with eight concepts, three property-introducing concepts and four intersection concepts

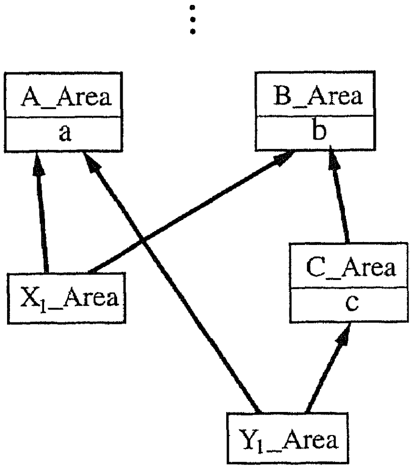


Figure 4.7 The OOVR schema for the CV shown in Figure 4.6

Figure 4.6 shows such an example. Concepts **A**, **B**, and **C** introduce attributes a, b, and c, respectively. Concepts $\mathbf{x}_1$, $\mathbf{x}_2$, $\mathbf{y}_1$, and $\mathbf{y}_2$ are intersection concepts (articulation concepts). Concepts with the same property set are in one block representing an area. The OOVR schema is shown in Figure 4.7. Via the SUBCLASS-_OF relationships, each class will have a correct property set for its instances. For example, class Y_1*_Area* has the property set $\{a,\ b,\ c\}$ and class X_1*_Area* has the property set $\{a,\ b\}$. Note that since $\mathbf{y}_3$ is not a root, X_1*_Area* is not a parent class of Y_1*_Area*. As we mentioned in the previous chapter, modeling a multi-rooted intersection area in such a way may pose problems when browsing the OOVR schema. For example, the concept level browsing path (**B**, $\mathbf{x}_1$, $\mathbf{y}_3$) has no parallel schema level browsing path since there is no link from Y_1*_Area* to X_1*_Area* to represent the link between concepts $\mathbf{x}_1$ and $\mathbf{y}_3$.

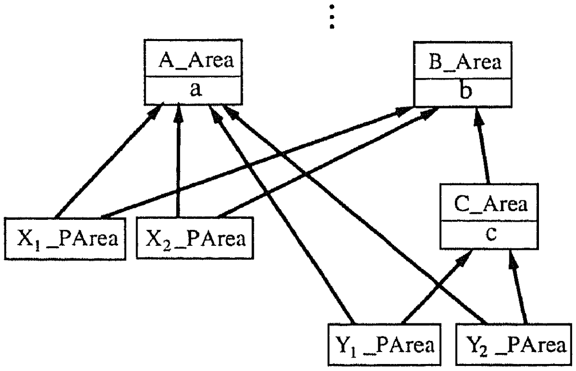


Figure 4.8 The singly-rooted schema for the CV shown in Figure 4.6

Figure 4.8 shows the singly-rooted schema for Figure 4.6. The intersection area X_1*_Area* has been partitioned into two p-areas: X_1*_PArea* and X_2*_PArea*. Similarly the intersection area Y_1*_Area* has been partitioned into two p-areas: Y_1*_PArea* and Y_2*_PArea*. This schema still reflects the problem of misleading users into thinking that between concepts of X_1*_PArea* and Y_1*_PArea* there are no IS-A relationships. In spite of the revised modeling there is still no schema level browsing path to the concept level browsing path (**B**, $\mathbf{x}_1$, $\mathbf{y}_3$). Since $\mathbf{x}_1$ is not in the same intersection area as y_3, the IS-A relationships between them will not be considered when the p-area diagram is generated. Therefore, concept y_3 will not be considered an articulation concept.

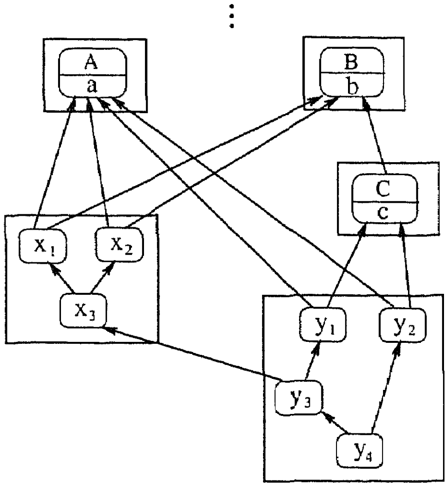


Figure 4.9 An example CV with ten concepts three property-introducing concepts and four intersection concepts

Figure 4.9 shows a more complicated, but interesting example. This example is the same as in Figure 4.6 except that there are two extra concepts, x_3 and y_4. Concept x_3 has two parents: x_1 and x_2; concept y_4 has two parents: y_2 and y_3. Concepts within a block have the same set of properties. Figure 4.7 shows the OOVR schema which did not change. The singly-rooted schema for this example is shown in Figure 4.10. We can see that compared to Figure 4.8, there are two extra p-areas, X_3_*PArea* and Y_4_*PArea*. This is because, according to our definition, concepts x_3 and y_4 are articulation concepts. However, concept y_3 is not an articulation concept because our methodology is bounded within a given intersection area. The partitioning of a multi-rooted intersection area has no effect or representing interconnection between its p-area to external areas or p-areas. Consequently, this singly-rooted schema will still mislead users into thinking that the concepts in X_3-_*PArea* and the concept in Y_1_*PArea* do not have IS-A relationships between them.

That, of course, is in this case, not true. Similar to the previous case, the concept level browsing path (**B**, $\mathbf{x}_2$, $\mathbf{x}_3$, $\mathbf{y}_3$) has no parallel schema level browsing path. Actually, Figure 4.10 has a more serious problem which does not appear in a simple CV like that of Figure 4.6. In this schema, there is no path between *X_3_PArea* and *Y_4_PArea*. The roots of these two p-area classes are $\mathbf{x}_3$ and $\mathbf{y}_4$, and they do have a path connecting them in the concept network. In other words, $\mathbf{y}_4$ is a descendant of $\mathbf{x}_4$. This missing path makes the singly-rooted schema modeling unsatisfactory. While in Figure 4.8, we did not have a schema level browsing path to a concept which is not a root of a p-area, in Figure 4.10, we do not have such a path even to a concept which is a root of a p-area.

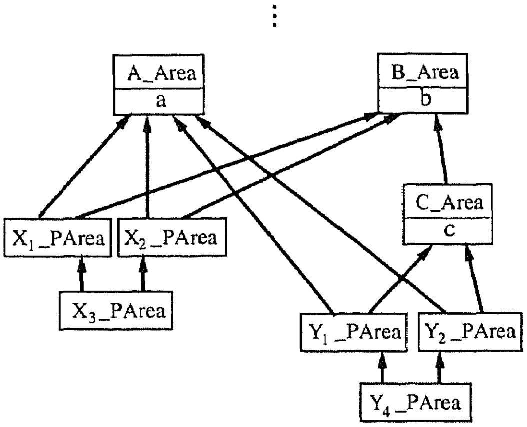


Figure 4.10 The singly-rooted schema for the CV shown in Figure 4.9

We can find many such examples in the singly-rooted MED OOVR schema. This kind of problem usually involves 4 to 5 area (and p-area) classes. Again, we start our explanation with the MED OOVR schema. Figure 4.11 shows such an example; it is an excerpt of the MED OOVR schema. The bottom two classes are multi-rooted intersection areas. Note that the class *Symptoms_Involving_Digestive_System_Area* is not a parent of class *Mental_Or_Behavioral_Dysfunction_Area*. That is because the concept **Mental or Behavioral Dysfunction** as well as the other roots of this area do not have parents residing in *Symptoms_Involving_Digestive_System_Area*.

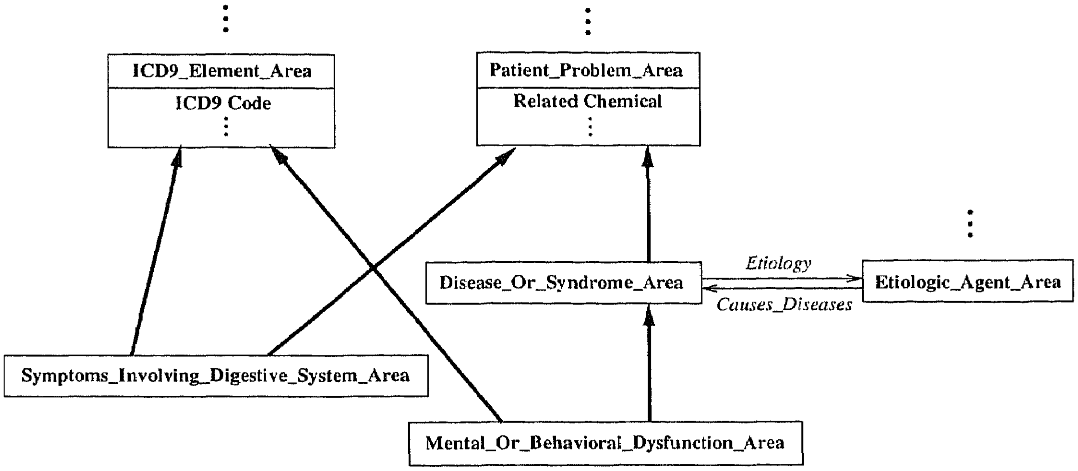


Figure 4.11 An excerpt of the MED OOVR schema

Figure 4.12 shows some concepts of the classes shown in Figure 4.11. For those two multi-rooted intersection areas, we show only some of their roots. The class *Mental_Or_Behavioral_Dysfunction_Area* has 29 roots in total, and the class *Symptoms_Involving_Digestive_System_Area* has 13 roots. Again, concepts within a block have the same property set. The concept **Unspecified Endocrine Disorder** is an ancestor (but not a parent) of the concept **Diabetes Mellitus** in this area. There are three concepts between them, and each one of them has only a single parent. The concept **Diabetes Mellitus** has another parent **Factor Influencing Health Status** residing in the *Symptoms_Involving_Digestive_System_Area*. The concept **Factor Influencing Health Status** is a root of its intersection area. Concept **Diabetic Myopathy** has two parents: **Diabetes Mellitus** and **Unspecified Disorder of Nervous System**. According to our revised mapping methodology, it will be considered an articulation concept.

If we apply our revised mapping methodology to the MED, the two intersection areas in Figure 4.12 will be partitioned into several p-areas. Figure 4.13 is an excerpt of the singly-rooted MED OOVR schema that reflects the concepts shown in Figure 4.12. Note that the concept **Diabetes Mellitus** will not be

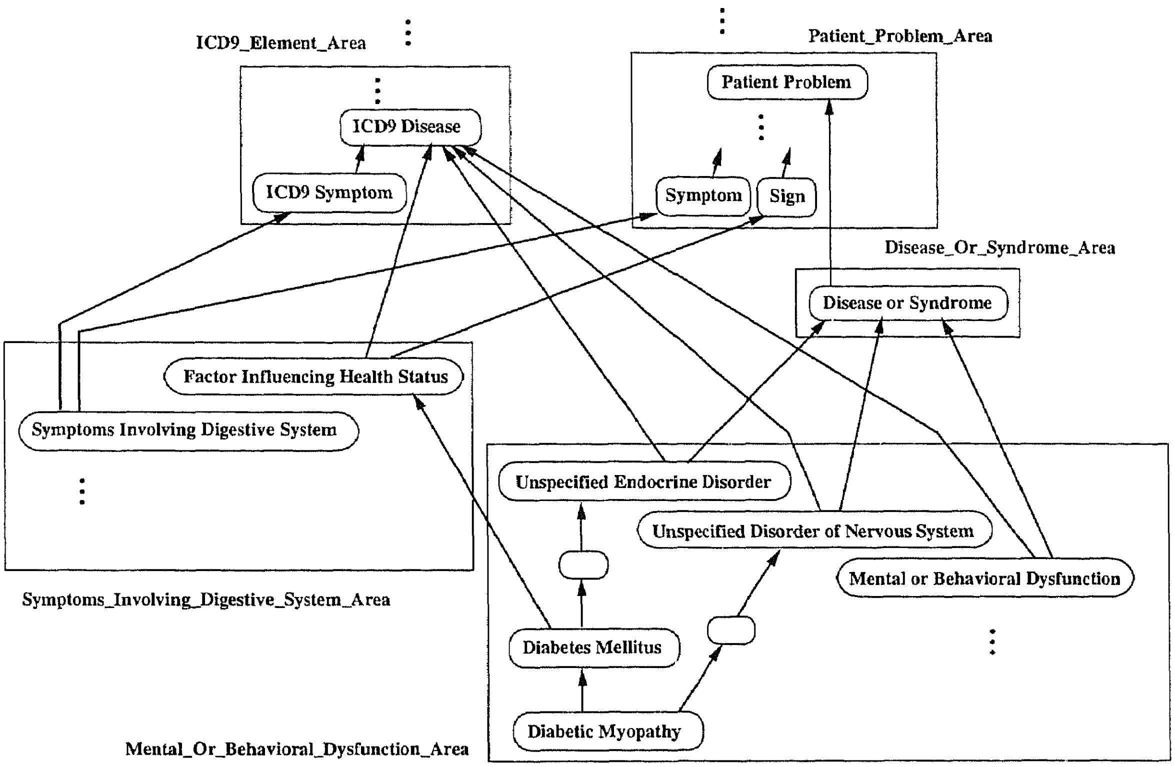

ICD9_Element_Area
ICD9 Disease
ICD9 Symptom
Patient_Problem_Area
Patient Problem
Symptom
Sign
Disease_Or_Syndrome_Area
Disease or Syndrome
Factor Influencing Health Status
Symptoms Involving Digestive System
Symptoms_Involving_Digestive_System_Area
Unspecified Endocrine Disorder
Unspecified Disorder of Nervous System
Mental or Behavioral Dysfunction
Diabetes Mellitus
Diabetic Myopathy
Mental_Or_Behavioral_Dysfunction_Area

Figure 4.12 The excerpt CV for the schema shown in Figure 4.11

considered an articulation concept since it has only one path to one intersection concept, **Unspecified Endocrine Disorder**, in its multi-rooted intersection area. The concept **Factor Influencing Health Status** is an intersection concept, but this fact does not have any effect when we further partition the class *Mental_Or-_Behavioral_Dysfunction_Area* in Figure 4.11. If **Diabetes Mellitus** is an articulation concept, then we will use its IS-A configuration to define the parent of its p-area. Unfortunately, because **Diabetes Mellitus** is not an articulation concept, the p-area it belongs to will not have any information to indicate the IS-A link between the concept **Diabetes Mellitus** and the concept **Factor Influencing Health Status**. Note that the property sets of the concept **Diabetes Mellitus** and the concept **Unspecified Endocrine Disorder** are identical and are a superset of the property set of the concept **Factor Influencing Health Status**. Furthermore, in Figure 4.13, there is no schema level browsing path between *Factor_Influencing-*

_Health_Status_PArea and *Diabetic_Myopathy_PArea.* That makes the singly-rooted schema unsatisfactory since the concept **Factor Influencing Health Status** is an ancestor of the concept **Diabetes Mellitus** and **Diabetic Myopathy**. The singly-rooted schema still cannot overcome this kind of problem.

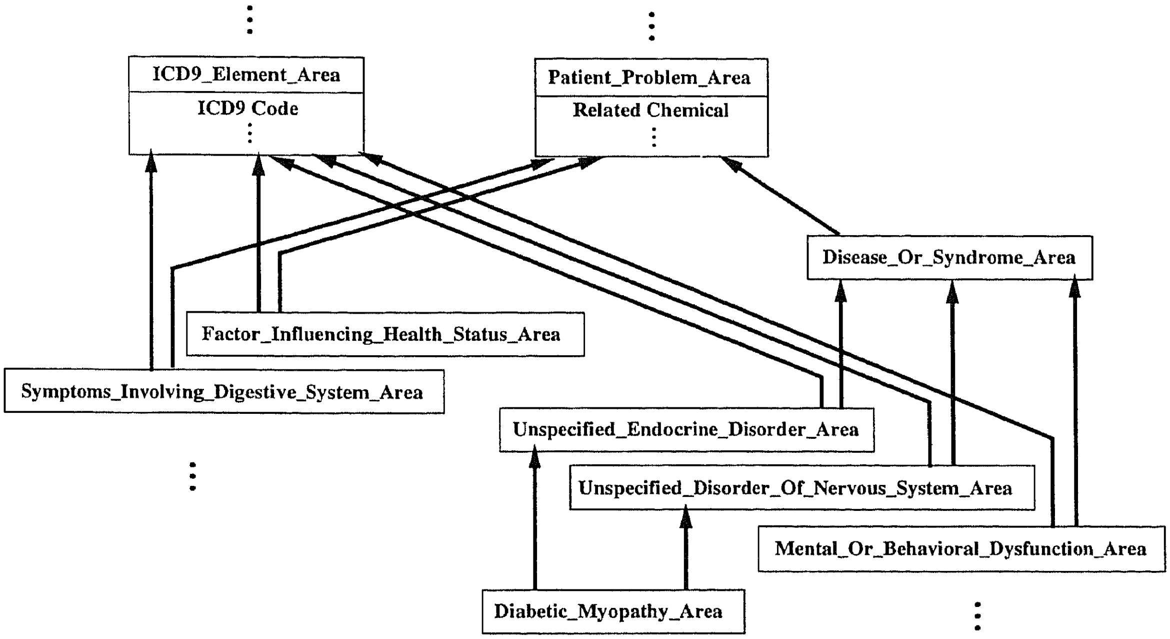

ICD9_Element_Area
ICD9 Code
Patient_Problem_Area
Related Chemical
Disease_Or_Syndrome_Area
Factor_Influencing_Health_Status_Area
Symptoms_Involving_Digestive_System_Area
Unspecified_Endocrine_Disorder_Area
Unspecified_Disorder_Of_Nervous_System_Area
Mental_Or_Behavioral_Dysfunction_Area
Diabetic_Myopathy_Area

Figure 4.13 An excerpt of the singly-rooted schema for the CV shown in Figure 4.12

Section 3.3 shows two problems of multi-rooted intersection areas. Those problems mainly occur when we browse the OOVR schema as an aid to comprehending the CV. Therefore, any missing SUBCLASS_OF relationship between two classes may lead to misunderstandings. The problems we present in this section are similar to the problem in Section 3.3.

4.2 IS-A$'$ Semantic Relationship

The three problems that were presented in the previous section, the removing of the short cut relationships, missing link problem, and partitioning boundary, arise from the fact that the SUBCLASS_OF relationships connecting area classes in a schema cannot fully reflect the IS-A relationships of all instances of a CV. Without a proper extra link to reflect IS-A relationships that are not captured by the SUBCLASS_OF relationships, the OOVR schema is a less effective abstraction. To overcome these problems, we define a new relationship IS-A$'$ for the singly-rooted schema. The SUBCLASS_OF relationships remain the same for the new schema.

Definition 15: (IS-A$'$ Relationship) Let X and Y be distinct area classes such that Y is not a SUBCLASS_OF X. If there exists a concept **c** in Y and a concept **d** in X such that **c** IS-A **d**, then we define Y IS-A$'$ X. □

As we mentioned before, the IS-A relationship in a CV has two functions. It is a mechanism for providing property inheritance. It also expresses specialization/generalization knowledge. The IS-A$'$ relationships in an OOVR schema reflect the second function of the IS-A relationships where it is not expressed by SUBCLASS_OF relationships.

The IS-A$'$ relationship is a hierarchical semantic relationship. It is hierarchical in nature, since it is derived from the hierarchical IS-A relationships of the CV. It will be noted that the SUBCLASS_OF relationships are also derived from the IS-A relationships of the CV. In contrast, when a class A is SUBCLASS_OF a class B then for every instance of A, there is an IS-A relationship or a chain of IS-A relationships in the CV leading to an instance of the class B. However, when A IS-A$'$ B, this only guarantees the existence of some instance or instances of A with an IS-A relationship to some instance(s) of B. As we see, IS-A$'$ models exceptional cases in the vocabulary, e.g., an IS-A relationship from a non-root instance of class A to a concept of class B where no root of A has such an IS-A relationship. Another example is when one of the roots of a multi-rooted intersection class does not have the connections to all the parent classes as the other roots. Thus, when the IS-A$'$ relationship is defined for a class, only some instances will have a value defined for this relationship. The IS-A$'$ relationships do not provide property inheritance between classes, as opposed to the SUBCLASS_OF relationships. The IS-A$'$ relationships do not satisfy transitivity, either. Our methodology performs the process of partitioning multi-rooted intersection classes into singly-rooted intersection classes for each such area separately. The result of the partitioning of a multi-rooted intersection area will not have any effect on the partitioning process for another multi-rooted intersection area. Otherwise, this process would be unnecessarily complex. Nevertheless the problem of missing links that interrupt necessary browsing paths is handled by adding the IS-A$'$ relationship at a second stage.

Users can navigate the OOVR schema using IS-A$'$ relationships as well. Hence, the definition of schema level browsing path can be extended to include the IS-A$'$ relationships as follows.

Definition 16: (Mixed Schema Level Browsing Path) A mixed schema level browsing path is a sequence of area classes (and p-area classes) $\mathrm{P}_A = (A_1, A_2, \ldots, A_k)$ such that A_{i+1} is SUBCLASS_OF A_i or A_{i+1} IS-A$'$ A_i. □

4.3 The Revised OOVR Schema with IS-A$'$ Relationships

We now apply the IS-A$'$ modeling to all the examples mentioned in Section 4.1. Figure 4.14 (b) shows the OOVR schema with IS-A$'$ relationships for the CV shown in Figure 4.14 (a). We use dashed arrows to represent IS-A$'$ in a diagram. The IS-A link from **z** to **x** in Figure 4.14 (a) induces the IS-A$'$ between *Z_Area* class and *W_Area* class in Figure 4.14 (b). The schema consisting of the SUBCLASS_OF relationships has no short cuts. The short cut from *Z_Area* class to *W_Area* class is captured by the IS-A$'$ relationship. The semantics represented by this IS-A$'$ relationship is that there exists a concept (**z**) in *Z_Area* and a concept (**x**) in *W_Area* such that in the CV these two concepts (**z** and **x**) are connected by an IS-A relationship. The other two IS-A relationships in the CV of Figure 4.14 (a), namely, **z** IS-A **y** and **y** IS-A

w, are represented by the two SUBCLASS_OF relationships which also support the inheritance of the attributes a and b. Consequently, the schema of Figure 4.14 (b) contains the mixed schema level browsing path (*W_Area*, *Z_Area*) parallel to the concept level browsing path (**w**, **x**, **z**).

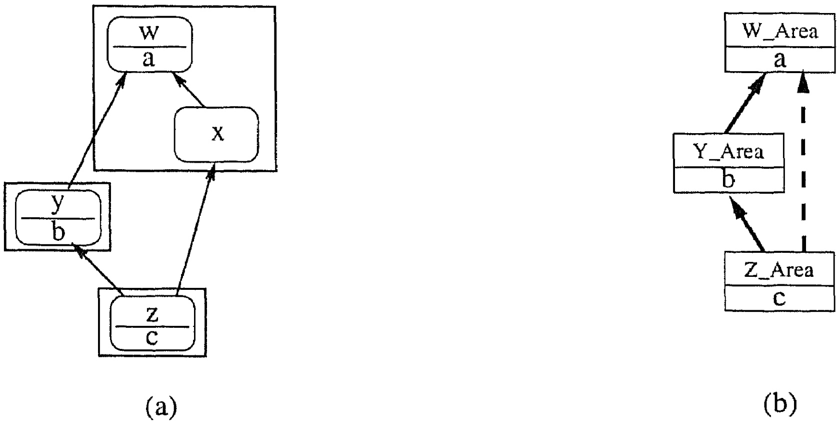


Figure 4.14 The singly-rooted schema with an IS-A$'$ relationship (b) for the CV in (a)

The missing link problem described in Section 4.1.2 can also be solved by adding IS-A$'$ relationships to the schema. Figure 4.15 (b) shows the singly-rooted schema for the CV of Figure 4.15 (a) with an IS-A$'$ relationship. Since *G_Area* IS-A$'$ *F_Area*, users are informed that some non-root concepts in *G_Area* have parents in *F_Area*. With this IS-A$'$ relationship, users can traverse from *G_Area* to *F_Area* or vice versa. The schema of Figure 4.15 (b) contains the mixed schema level browsing path (*D_Area*, *F_Area*, *G_Area*) parallel to the concept level browsing path (**D**, **F**, **H**).

The problem of the partitioning boundary described in Section 4.1.3 can also be solved by the IS-A$'$ relationships. That is because, by definition, IS-A$'$ relationships are not restricted to one multi-rooted intersection area. In other words, such a relationship can cross the boundary of a multi-rooted intersection area. Figure 4.16 (b) shows the singly-rooted schema with IS-A$'$ relationships for

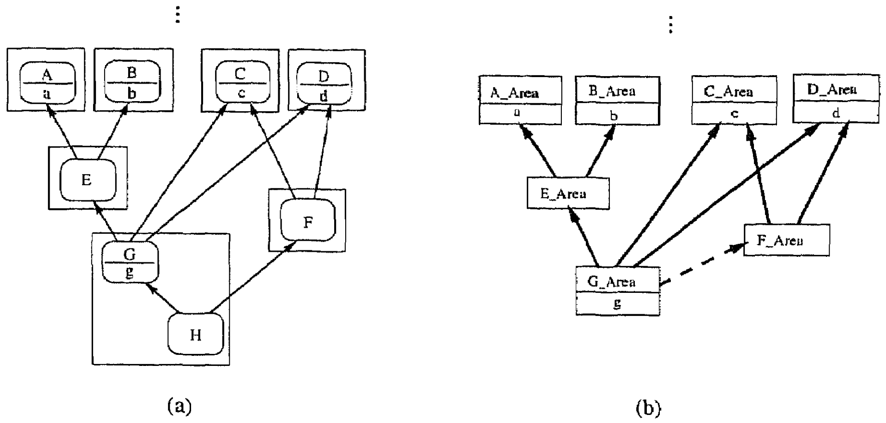


Figure 4.15 The singly-rooted schema with an IS-A$'$ relationship (b) for the CV in (a)

the CV shown in Figure 4.16 (a). In Figure 4.16 (b), all the links between classes are SUBCLASS_OF relationships, except for the IS-A$'$ between X_1_*Area* and Y_1_*Area*. With this relationship, users are made aware that there exists at least one pair of concepts in these two classes connected by an IS-A relationship. For instance, the schema of Figure 4.15 (b) contains the mixed schema level browsing path (B_*Area*, X_1_*Area*, Y_1_*Area*) parallel to the concept level browsing path (**B**, $\mathbf{x}_1$, $\mathbf{y}_3$).

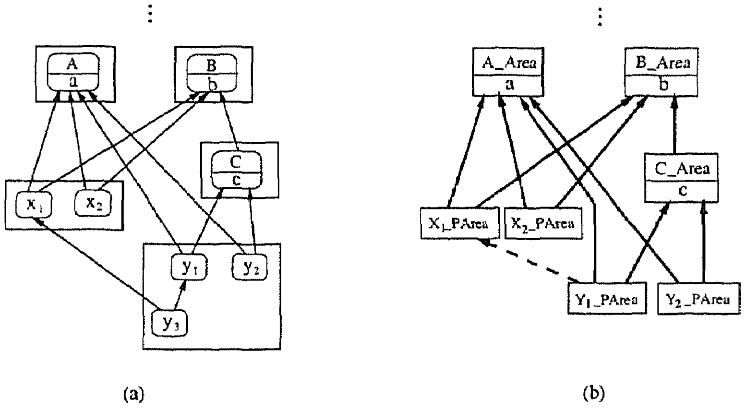


Figure 4.16 The singly-rooted schema with IS-A$'$ relationships (b) for the CV in (a)

Figure 4.17 (b) shows the singly-rooted schema with IS-A$'$ relationships for the CV shown in Figure 4.17 (a). In this figure, there is only one IS-A$'$ relationship, the one between X_3_*Area* and Y_1_*Area*. From this example, we can see how important an IS-A$'$ relationship can be. Since concept $\mathbf{y}_1$ is not a descendant of $\mathbf{x}_3$, it may look correct to have no relationship between Y_1_*Area* and X_3_*Area*. However, concept $\mathbf{y}_4$ is not only a descendant of $\mathbf{y}_1$ but also a descendant of $\mathbf{x}_3$. Without a link between Y_1_*Area* and X_3_*Area*, the schema is deficient because it has no paths between Y_4_*Area* and X_3_*Area*. The IS-A$'$ relationship between Y_1_*Area* and X_3_*Area* makes up for the missing link and creates a mixed path composed of SUBCLASS_OF and IS-A$'$ relationships between Y_4_*Area* and X_3_*Area*. For instance, the schema of Figure 4.16 (b) contains the mixed schema level browsing path (*B_Area*, X_1_*Area*, X_3_*Area*, Y_1_*Area*, Y_4_*Area*) parallel to the concept level browsing path (**B**, $\mathbf{x}_1$, $\mathbf{y}_3$, $\mathbf{y}_4$).

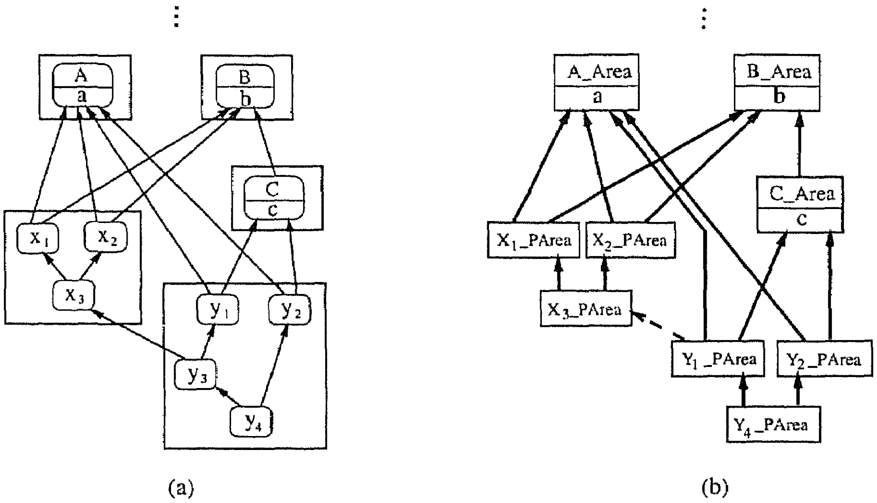


Figure 4.17 The singly-rooted schema with an IS-A$'$ relationship (b) for the CV in (a)

Figure 4.18 (b) shows the singly-rooted schema with IS-A$'$ relationships for the CV shown in Figure 4.18 (a). The IS-A link between **Diabetes Mellitus** and **Factor Influencing Health Status** is represented by the IS-A$'$ relationship between *Unspecified_Endocrine_Disorder_Area* and *Physical_Finding_Area*. This IS-A$'$ relationship represents an important connection between *Factor_Influencing-*

_Health_Status_PArea and *Diabetic_Myopathy_PArea*. Without it, the singly-rooted schema is incorrect since **Diabetic Myopathy** IS-A **Factor Influencing Health Status** is not reflected. In other words, the concept level browsing path (**Factor Influencing Health Status**, **Diabetes Mellitus**, **Diabetic Myopathy**) has a parallel mixed schema level browsing path (*Factor_Influencing_Health_Status_PArea*, *Unspecified_Endocrine_Disorder_Area*, *Diabetic_Myopathy_PArea*).

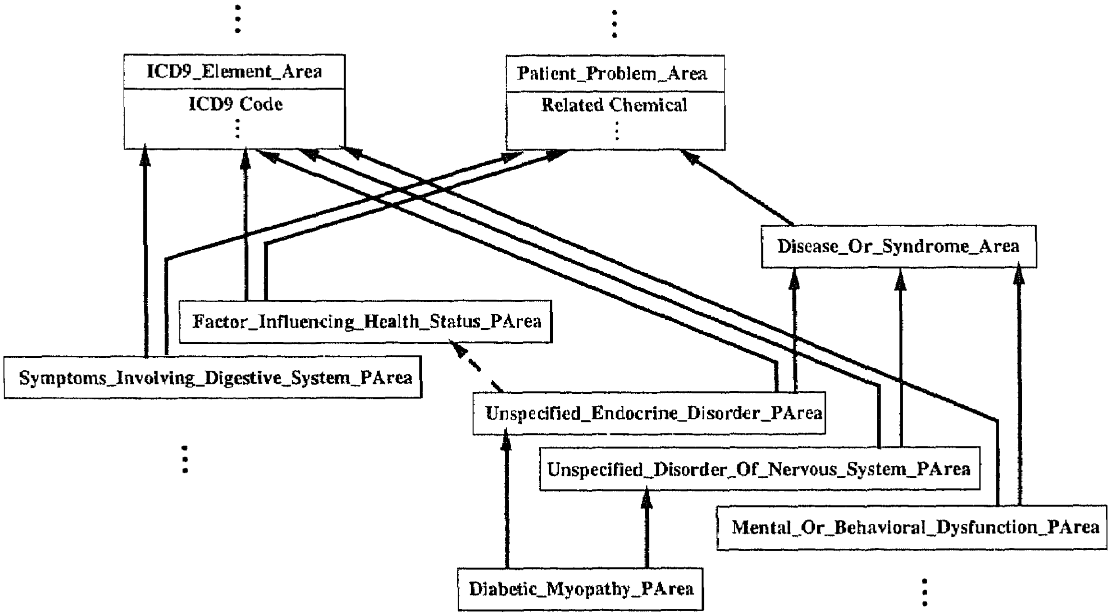

ICD9_Element_Area
ICD9 Code
Patient_Problem_Area
Related Chemical
Disease_Or_Syndrome_Area
Factor_Influencing_Health_Status_PArea
Symptoms_Involving_Digestive_System_PArea
Unspecified_Endocrine_Disorder_PArea
Unspecified_Disorder_Of_Nervous_System_PArea
Mental_Or_Behavioral_Dysfunction_PArea
Diabetic_Myopathy_PArea

Figure 4.18 The singly-rooted schema with IS-A$'$ relationships for the CV in Figure 4.12

With the IS-A$'$ relationships, the singly-rooted schema not only solves the problems described in Section 4.1, but also exhibits several interesting properties. In the next section, we prove some theorems pertaining to the singly-rooted schema with IS-A$'$ relationships. To summarize, with the addition of an IS-A$'$ relationship in every place where a link was missing in the schema, we create a mixed schema level browsing path for every concept level browsing path in the CV.

4.4 Properties of a Revised OOVR Schema with IS-A$'$ Relationships

In this section, we exhibit some interesting theoretical characteristics of the singly-rooted OOVR representation. We will be stating a number of these as theorems and lemmas, along with proofs. The family terms (parent, child, ancestor, descendant) used between two areas are induced from the SUBCLASS_OF relationships on the singly-rooted Schema. For instance, if class A is a SUBCLASS_OF class B, then we say that A is a parent of B. If A is a descendant of B, then there exists a path of SUBCLASS_OF relationships between A and B.

Definition 17: (Completely Direct Articulation Ancestor [CDAA]) Let $\mathbf{v}$ be a concept and $\mathbf{w}$ be an articulation concept such that $\mathrm{Desc}(\mathbf{v}, \mathbf{w})$ and $\mathbf{v}$ and $\mathbf{w}$ are in the same multi-rooted intersection area [i.e., $\mathrm{P}(\mathbf{v}) = \mathrm{P}(\mathbf{w}]$. The concept $\mathbf{w}$ is called a Completely Direct Articulation Ancestor (or CDAA) of $\mathbf{v}$ if there are no paths from $\mathbf{v}$ to $\mathbf{w}$ that contain other articulation concepts. □

Lemma 9: A concept $\mathbf{v}$ in a multi-rooted intersection area is an articulation concept if and only if $\mathbf{v}$ has no CDAA or more than one CDAA.

Proof:

($\Rightarrow$): Let $\mathbf{v}$ be an articulation concept. If $\mathbf{v}$ is an intersection concept, then it has no CDAA since it is a root. If $\mathbf{v}$ is a non-root articulation concept, by Definition 10 (a), it must have two articulation ancestors $\mathbf{x}$ and $\mathbf{y}$ and by Definition 10 (b), no other articulation concepts exist on the paths between $\mathbf{v}$ to $\mathbf{x}$ and $\mathbf{v}$ to $\mathbf{y}$. By Definition 17, $\mathbf{x}$ and $\mathbf{y}$ are CDAAs of $\mathbf{v}$. In other words, $\mathbf{v}$ has at least two CDAAs.

($\Leftarrow$): Let $\mathbf{v}$ be an concept with zero or more than one CDAA in an multi-rooted intersection area A.

Case 1: Assume $\mathbf{v}$ has no CDAA. If $\mathbf{v}$ is a root of A, then by Definition 10 (a), $\mathbf{v}$ is an articulation concept. Otherwise, $\mathbf{v}$ must have some articulation ancestors. Due to the fact that a CV is acyclic, **v** must have a CDAA among its articulation ancestors. Case 2: Assume a concept **v** in area A has more than one CDAA. By Definition 10 (b), **v** is an articulation concept. ■

Lemma 10: Let **p** and **q** be articulation ancestors of a non-articulation concept **x**. If **q** is an ancestor of **p**, then **x** does not belong to the p-area rooted at **q**.

For a CV **V**, this lemma can be summarized as follows:

$$\forall \mathbf{x}, \mathbf{p}, \mathbf{q} \in \mathbf{V} \text{ s.t. } \mathbf{p} \text{ and } \mathbf{q} \text{ (but not } \mathbf{x}\text{) are articulation concepts}$$

$$(\mathrm{Desc}(\mathbf{x}, \mathbf{p}) \wedge \mathrm{Desc}(\mathbf{x}, \mathbf{q}) \wedge \mathrm{Desc}(\mathbf{p}, \mathbf{q})) \Rightarrow \mathbf{x} \notin \mathrm{PArea}(\mathbf{q})$$

Proof:

Case 1: **p** and **q** have the same set of properties.

Case 1.1: If no articulation concept exists between **p** and **q**, then **p** is a DARD of **q**. By Definition 12, **x** will be excluded from the p-area rooted at **q** by **p**.

Case 1.2: If articulation concepts exist between **p** and **q**, then (at least) one of them **m** is a DARD of **q**, and **m** is also an ancestor of **x**. Therefore, by Definition 12, **x** will be excluded from the p-area rooted at **q** by **m**.

Case 2: **p** and **q** have different property sets [i.e., $\mathrm{P}(\mathbf{p}) \neq \mathrm{P}(\mathbf{q})$].

$\mathrm{Desc}(\mathbf{p}, \mathbf{q})$ implies that $\mathrm{P}(\mathbf{q}) \subseteq \mathrm{P}(\mathbf{p})$. Since $\mathrm{P}(\mathbf{p}) \neq \mathrm{P}(\mathbf{q})$, $\mathrm{P}(\mathbf{q}) \subset \mathrm{P}(\mathbf{p})$. $\mathrm{Desc}(\mathbf{x}, \mathbf{p})$ implies that $\mathrm{P}(\mathbf{p}) \subseteq \mathrm{P}(\mathbf{x})$. Applying transitivity to $\mathrm{P}(\mathbf{q}) \subset \mathrm{P}(\mathbf{p})$ and $\mathrm{P}(\mathbf{p}) \subseteq \mathrm{P}(\mathbf{x})$, we have $\mathrm{P}(\mathbf{q}) \subset \mathrm{P}(\mathbf{x})$, i.e., **q** and **x** have different sets of properties. In a singly-rooted schema, concepts within a p-area (p-area class) have the same set of properties. Therefore, **x** cannot be in the area rooted at **q**. ■

Lemma 11: A p-area cannot have IS-A$'$ relationships pointing to its descendant p-areas.

Proof: Assume to the contrary that a p-area X IS-A$'$ Y where Y is a descendant of X. Since X IS-A$'$ Y, there exist concepts $\mathbf{v} \in X$ and $\mathbf{w} \in Y$ such that $\mathbf{v}$ IS-A $\mathbf{w}$. Let the roots of X and Y be $\mathbf{r}_X$ and $\mathbf{r}_Y$, respectively. Desc($\mathbf{r}_Y$, $\mathbf{r}_X$) since Y is a descendant of X.

If $\mathbf{w} = \mathbf{r}_Y$, $\mathbf{v}$ IS-A $\mathbf{w}$ implies Desc($\mathbf{v}$, $\mathbf{r}_Y$). In case of $\mathbf{w} \neq \mathbf{r}_Y$, since $\mathbf{w} \in Y$, Desc$(\mathbf{w}, \mathbf{r}_Y)$ holds. By transitivity of $\mathbf{v}$ IS-A $\mathbf{w}$ and Desc($\mathbf{w}$, $\mathbf{r}_Y$), we get Desc($\mathbf{v}$, $\mathbf{r}_Y$).
Case 1: $\mathbf{v} = \mathbf{r}_X$: Desc($\mathbf{r}_X$, $\mathbf{r}_Y$) holds. However, we showed before that Desc($\mathbf{r}_Y$, $\mathbf{r}_X$) holds. Together they imply a cycle in the CV—a contradiction.
Case 2: $\mathbf{v} \neq \mathbf{r}_X$: Desc($\mathbf{v}$, $\mathbf{r}_X$) holdS. As we showed before, Desc($\mathbf{v}$, $\mathbf{r}_Y$) and Desc($\mathbf{r}_Y$, $\mathbf{r}_X$) hold. However, by Lemma 10, if a concept $\mathbf{v}$ has two articulation ancestors $\mathbf{r}_X$ and $\mathbf{r}_Y$ such that Desc$(\mathbf{v}, \mathbf{r}_X)$, Desc$(\mathbf{v}, \mathbf{r}_Y)$, and Desc$(\mathbf{r}_Y, \mathbf{r}_X)$, then $\mathbf{v} \notin X$—a contradiction. ■

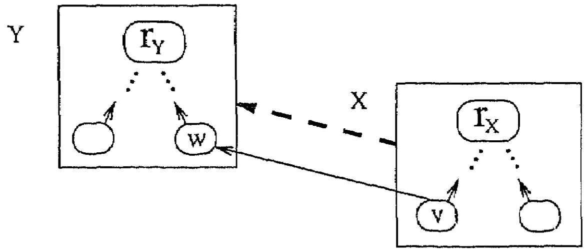


Figure 4.19 A CV with six concepts

Two areas X and Y are called *independent* if $X \neq Y$ and one is not descendant of the other.

Lemma 12: There is no IS-A$'$ relationship connecting two independent p-areas of the same multi-rooted intersection area.

Proof: Assume to the contrary that p-area X IS-A$'$ Y, such that X and Y are independent p-areas of the same multi-rooted intersection area. That means that there is a concept $\mathbf{v}$ in p-area X (rooted at $\mathbf{r}_X$) which has a parent residing in Y (rooted at $\mathbf{r}_Y$), see Figure 4.19.

Case 1: $\mathbf{v} = \mathbf{r}_X$: By the definition of the SUBCLASS_OF relationship either, X is a SUBCLASS_OF Y, or this SUBCLASS_OF relationship was removed as a short cut since X is a descendant of Y. Both cases contradict the independence of X and Y.

Case 2: $\mathbf{v} \neq \mathbf{r}_X$:

Case 2.1: If $\mathbf{r}_Y$ is a CDAA of $\mathbf{v}$, then $\mathbf{v}$ has at least two CDAAs, $\mathbf{r}_Y$ and $\mathbf{r}_X$. By Lemma 9, $\mathbf{v}$ must be an articulation concept—a contradiction, since $\mathbf{r}_X$ is the only articulation concept of X.

Case 2.2: If $\mathbf{r}_Y$ is not a CDAA of $\mathbf{v}$, then there exists an articulation concept $\mathbf{r}_P$ in P on one of the paths between $\mathbf{v}$ and $\mathbf{r}_Y$, which is a CDAA of $\mathbf{v}$.

By Case 2.1 replacing $\mathbf{r}_Y$ by $\mathbf{r}_P$, we have a contradiction. ■

Lemmas 11 and 12 together give us:

Theorem 6: Let X and Y be two p-areas in the same multi-rooted intersection area with X IS-A$'$ Y. Then Y must be an ancestor of X.□

Since the IS-A$'$ relationships are induced by the hierarchical IS-A relationships in a CV, they are also hierarchical relationships. Thus, there should be no cycle of IS-A$'$ and SUBCLASS_OF relationships in the singly-rooted schema similar to Theorem 5 which we proved in Chapter 3, regarding only the SUBCLASS_OF relationships.

Theorem 7: In a singly-rooted schema, no IS-A$'$ or SUBCLASS_OF cycle can exist within a multi-rooted intersection area.□

Proof: Let $\mathcal{R}$ denote either a SUBCLASS_OF or an IS-A$'$ relationship. Assume to the contrary that there does exist a cycle containing m ($m > 1$) p-areas $Z_1, Z_2, \ldots, Z_m$ such that $Z_1 \mathcal{R} Z_2$, $Z_2 \mathcal{R} Z_3, \ldots, Z_{m-1} \mathcal{R} Z_m$, and $Z_m \mathcal{R} Z_1$ (see Figure 4.20).

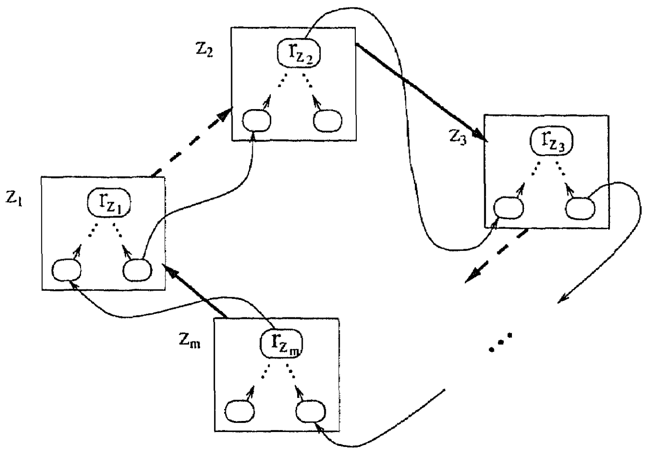


Figure 4.20 A cycle of SUBCLASS_OF and IS-A$'$ relationships can not occur in a singly-rooted schema.

Let Z_i $\mathcal{R}$ Z_{i+1} $1 \leq i < m$. Either Z_i SUBCLASS_OF Z_{i+1} or Z_i IS-A$'$ Z_{i+1}, in which case by Theorem 6 Z_{i+1} is a ancestor pf Z_i. That is, there is a path of SUBCLASS_OF relationship from Z_i to Z_{i+1}. Hence, there exists a path of SUBCLASS_OF relationships from Z_1 to Z_m. But there exists also a path of one or more SUBCLASS_OF relationships from Z_m to Z_1 since Z_m $\mathcal{R}$ Z_1. A contradiction to Theorem 5. ■

It is important to prove that the singly-rooted schema does not contain cycles consisting of these relationships, since both SUBCLASS_OF and IS-A$'$ are hierarchical relationships reflecting IS-A semantics on a CV, and a CV is a DAG. Theorem 5 states that there is no SUBCLASS_OF cycle on a singly-rooted schema, and Theorem 7 proves that there is no IS-A$'$ or SUBCLASS_OF cycle among p-areas of a multi-rooted intersection area class. In the following, we prove that there is no cycle composed of these two relationships in the singly-rooted schema.

Theorem 8: In a singly-rooted schema, no SUBCLASS_OF or IS-A$'$ cycle can exist. □

Proof: Let $\mathcal{R}$ be either a SUBCLASS_OF or an IS-A$'$ relationship. Assume to the contrary that there exists such a cycle containing m classes (area or p-area classes)

$Z_1, Z_2, \ldots, Z_m$ $(m > 1)$ with $Z_1 \mathcal{R} Z_2$, $Z_2 \mathcal{R} Z_3, \ldots, Z_{m-1} \mathcal{R} Z_m$, and $Z_m \mathcal{R} Z_1$ (See Figure 4.20).

For two successive classes Z_i and Z_{i+1} $1 \leq i < m$, if they are part of the same multi-rooted intersection area, their property sets are the same, i.e., $\mathrm{P}(Z_i)$=$\mathrm{P}(Z_{i+1})$. If they are not from the same multi-rooted intersection area, Z_i must have more properties than Z_{i+1} since, by definition, **q** IS-A **p** for some $\mathbf{q} \in Z_i$ and $\mathbf{p} \in Z_{i+1}$ and thus **q** inherits the properties of **p**. In other words, $\mathrm{P}(Z_i) \supset \mathrm{P}(Z_{i+1})$. In general, $\mathrm{P}(Z_i) \supseteq \mathrm{P}(Z_{i+1})$. Thus, $\mathrm{P}(Z_1) \supseteq \mathrm{P}(Z_2) \supseteq \mathrm{P}(Z_3) \supseteq \cdots \supseteq \mathrm{P}(Z_m)$. By transitivity, $\mathrm{P}(Z_1) \supseteq \mathrm{P}(Z_m)$. On the other hand, $\mathrm{P}(Z_m) \supseteq \mathrm{P}(Z_1)$ since $\mathrm{P}(Z_m)$ IS-A$'$ $\mathrm{P}(Z_1)$. Hence: $\mathrm{P}(Z_1) = \mathrm{P}(Z_2) = \mathrm{P}(Z_3) = \cdots = \mathrm{P}(Z_m) = \mathrm{P}(Z_1)$, which means that the cycle must be confined to a single intersection area. By Theorem 7, we know that this is impossible. ■

Lemma 13: Let X and Y be two p-area classes in the same multi-rooted intersection area with roots $\mathbf{r}_X$ and $\mathbf{r}_Y$, respectively. If concept $\mathbf{r}_X$ is a CDAA of $\mathbf{r}_Y$ then Y is a SUBCLASS_OF X.

Proof: If concept $\mathbf{r}_X$ is a CDAA of $\mathbf{r}_Y$ then there is no path between $\mathbf{r}_X$ and $\mathbf{r}_Y$ containing other articulation concepts. Because of the way we create the SUBCLASS-_OF relationships, described in Section 3.4.3, Y is a SUBCLASS_OF X. ■

The following theorem reflects our philosophy of defining articulation concepts. In Figure 4.21, concept $\mathbf{r}_Y$ is an ancestor of $\mathbf{r}_X$ and all other parent concepts of $\mathbf{r}_X$ are in the parent areas of Y. Therefore, instead of making $\mathbf{r}_X$ an articulation concept, we would like to group $\mathbf{r}_X$ into the p-area rooted at $\mathbf{r}_Y$. The definition of articulation concept follows this philosophy.

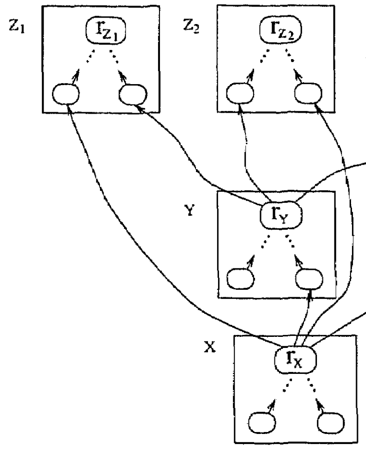


Figure 4.21 An example where area X will be merged into area Y

Theorem 9: In a given multi-rooted intersection area, no p-area X having $m+1$ parent p-areas ($m > 0$) Y, Z_1, $Z_2, \ldots$, Z_m can exist such that Y is a parent of X and descendant of all Z_i's. □

Proof: Assume to the contrary that X has $m+1$ parent p-areas ($m > 0$) Y, Z_1, $Z_2, \ldots$, Z_m rooted at $\mathbf{r}_Y$, $\mathbf{r}_{Z_1}$, $\mathbf{r}_{Z_2}, \ldots$,$\mathbf{r}_{Z_m}$, respectively, where Y is a parent of X and descendant of all Z_i's (see Figure 4.21). Let $\mathbf{r}_X$ be the root of X. Note that $\mathbf{r}_X$ cannot be an intersection concept since it has articulation ancestors $\mathbf{r}_Y$ and Z_i $(1 \le i \le m)$.

Concept $\mathbf{r}_X$ is a descendant of $\mathbf{r}_Y$, $\mathbf{r}_{Z_1}$, $\mathbf{r}_{Z_2}, \ldots$, $\mathbf{r}_{Z_m}$, since X is a descendant of Y, Z_1, $Z_2, \ldots$, Z_m. Similarly, $\mathbf{r}_Y$ is a descendant of $\mathbf{r}_{Z_1}$, $\mathbf{r}_{Z_2}, \ldots$, $\mathbf{r}_{Z_m}$, since Y is a descendant of Z_1, $Z_2, \ldots$,Z_m. Therefore, $\mathbf{r}_{Z_1}$, $\mathbf{r}_{Z_2}, \ldots$, $\mathbf{r}_{Z_m}$ cannot be CDAAs of $\mathbf{r}_X$, because $\mathbf{r}_Y$ is on paths between $\mathbf{r}_X$ and all $\mathbf{r}_{Z_i}$'s. By Lemma 13, $\mathbf{r}_X$'s CDAAs must reside within Y, Z_1, $Z_2, \ldots$, Z_m. More specifically, the only possible CDAAs for $\mathbf{r}_X$ are $\mathbf{r}_Y$, $\mathbf{r}_{Z_1}$, $\mathbf{r}_{Z_2}, \ldots$, $\mathbf{r}_{Z_m}$. However, since all $\mathbf{r}_{Z_i}$ $(1 \le i \le m)$ are not CDAAs of $\mathbf{r}_X$, $\mathbf{r}_Y$ is the only possible CDAA of $\mathbf{r}_X$. By Lemma 9, $\mathbf{r}_X$ cannot be an articulation concept with less than two CDAAs, since it is not an intersection concept—a contradiction.

■

CHAPTER 5

THE MULTILEVEL AREA DIAGRAM

In Chapter 2 and Chapter 3, we presented techniques to generate two abstractions of a CV: the OOVR schema and the singly-rooted schema. Chapter 4 presented some problems with the OOVR schema and their solutions in the form of the singly-rooted schema. We can choose either one of these schemas to store concepts in an OODB and to provide an abstraction of the CV. Since the OOVR schema is used not only for storage purposes but also for capturing the abstraction of a CV, users may want to have both schemas available at the same time. Unfortunately, it is impossible to have two schemas for one OODB.

For example, the MED OOVR[1] schema has 124 area classes. The singly-rooted MED OOVR schema has 1,835 classes. Under the OODB framework, browsing is limited to only one of these two. For example, the following scenario is not allowed. Assume a user browses the original OOVR schema first. If he reaches a complicated multi-rooted intersection area, then he may want to switch to the singly-rooted schema to get a more detailed abstraction of the CV. If he continues to browse the singly-rooted schema and feels it is too detailed, he may want to switch back to the more general multi-rooted OOVR schema. This scenario is possible only if we have both schemas (or abstraction networks) on hand simultaneously. The current OODB model does not support this kind of browsing. If a user browses the singly-rooted schema, then no “multi-rooted intersection area class” is available. If a user browses the singly-rooted schema, no “p-area” can be accessed.

In this chapter, we propose a new model, called a multilevel area diagram, to represent the abstraction of a CV. Using area diagrams, we will define the multilevel area diagram model which has two levels of abstraction on top of the instance level.

[1] In previous chapters, the MED refers to the CPMC MED 1996 version. In this chapter, we are using the 1998 version with over 56,000 concepts.

In this way, users will have more than one choice with respect to their browsing and comprehension needs. In short, the new model will offer the users the best of both worlds, the compactness of the OOVR schema and the detailed modeling of multi-rooted intersection areas as singly-rooted p-areas in the singly-rooted schema.

5.1 Areas and Area Partitions

Similar to an OODB schema, an area diagram (defined below) can be used as an abstraction of a CV. The notions of area and area partition are defined as follows.

In general, an area should capture a unified subhierarchy of concepts. However, we cannot require such subhierarchies to be connected, since in the case of a multi-rooted area diagram the subhierarchies rooted at each root may be disconnected from each other. Thus, we limit the requirement to *continuity* along the hierarchical dimension, called hierarchical continuity, which we define as follows.

Definition 18: (Area) Let **V** be a CV. An area A is a non-empty set of concepts such that the following two conditions are satisfied.

1. Property uniformity: concepts within an area have the same sets of properties, i.e., $\forall x, y \in A, \mathrm{P}(x) = \mathrm{P}(y)$.
2. Hierarchical continuity: $\forall$ **x**, **y** $\in A$ such that there exists a path of IS-A relationships in the CV from **x** to **y**, all the concepts along this path also belong to A. □

Note that the hierarchical continuity disallows an area A as shown in Figure 5.1.

Definition 19: (Area Partition) Let **V** be a CV. An area partition **PR(V)** is a set of areas $\{X_i \mid 1 \leq i \leq k\}$ such that $\forall X_i, X_j$ $(1 \leq i, j \leq k$ and $i \neq j)$ $\in$ $\mathbf{PR(V)}, X_i \cap X_j = \emptyset$ and $\bigcup_{i=1}^{k} X_i = \mathbf{V}$. By definition, an area partition is a partition of a CV. **PR(V)** can be denoted as (**V**, f) since with the function f defined for the CV **V**, we are able to derive the proper area partition for a concept v by f(v). □

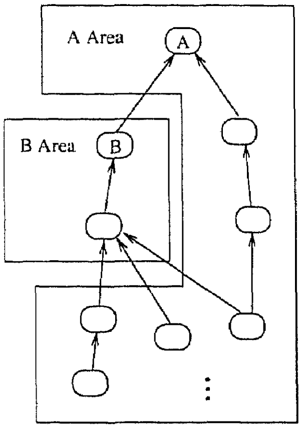


Figure 5.1 An illegal grouping of concepts which violates hierarchical continuity

The results of both the original and revised partitioning criteria are area partitions. Both these area partitions satisfy Definitions 18 and 19. In addition, the original area diagram and the singly-rooted area diagram satisfy the following special conditions. In the original area diagram, an area is maximal in that it contains all concepts with a given set of properties. Formally, an area A in a CV **V** is maximal if $\forall$ **x** $\in$ **V**, if P(**x**) = P(A), then **x** $\in A$. The singly-rooted area diagram has the property that each area has a single root. Formally, $\forall A$, $\exists$ **r** $\in A$ such that **r** is a root and $\forall$ **x** $\in A$ (where **x** $\neq$ **r**), **x** is a descendant of **r**. As mentioned in earlier sections, this property of the singly-rooted area diagram guarantees uniform semantics of its concepts.

In this chapter, we do not specify how to create an area or an area partition. Instead, we only address the constraints of areas and area partitions. The techniques described in Chapter 2 and Chapter 3 will generate the original area partition and the singly-rooted area partition of a CV, respectively.

If an area is singly-rooted, then the root **C** will be the naming concept of this area. The area name is created by adding the suffix "Area" to the name of the naming concept. Hence, we will call this area *C Area*. Since an area name is not a class name in an OODB, it can contain spaces or any other special characters. If an area is multi-rooted, we randomly select one root to name this area with an asterisk followed by the number of roots within parentheses, and the suffix "Area."

5.2 Area Diagrams

Once we partition a CV **V** into an area partition **PR(V)**, we need to define an area diagram which also represents the relationships between pairs of areas in **PR(V)**. Similar to the IS-A relationship in a CV, the relationships between areas should perform two functions: property inheritance and subsumption between areas. The SUBCLASS_OF relationship is not proper for area diagrams since an area diagram is not an OODB schema. In Chapter 6, when discussing open problems, we will mention the issue of a system with more than two area diagrams.

Definition 20: (Area Diagram) Let **V** be a CV. An area diagram **AD(V)** is an area partition **PR(V)** with appropriate child_of and IS-A$'$ relationships defined between areas in **PR(V)**. □

In the rest of this section, we will define the two kinds of relationships child_of and IS-A$'$ in detail. We define the relationship child_of to capture the hierarchical structure of the areas in an area diagram.

Let **v** be a concept in a CV and Prn(**v**) be the set of **v**'s parents. As we defined before, Area(**v**) is the area containing concept **v**. Let AreaPrn(**v**) be the set of areas of v's parents, i.e., AreaPrn(**v**) = { Area(**x**) | **x** ∈ Prn(**v**) }.

We now define the child_of relationship.

Definition 21: **(Child_of)** Let X be an area with n roots $\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_n$. Let $\mathcal{S}$ be a set of areas where $\mathcal{S} = \bigcap_{i=1}^{n} \text{AreaPrn}(\mathbf{r_i})$. Then X is a child_of $\mathcal{S}$. In other words, if X is a child_of Y, then all roots of X have at least one parent concept in Y. If X is a child_of Y, we say that Y is a parent of X. □

According to Definition 21, if an area X is singly-rooted at $\mathbf{r}$, the parents of X are induced by the IS-A configuration of $\mathbf{r}$. Assume that $\mathbf{r}$ has n parents residing in m areas. Those m areas will be the parents of X. On the other hand, if X has l ($l > 1$) roots $\mathbf{r}_1, \mathbf{r}_2, \ldots, \mathbf{r}_l$, then the definition of X's parents is more subtle. An area Y is said to be a parent of X if for all $\mathbf{r}_i$ $(1 < i < l)$ there exist concepts $\mathbf{y}_j$ $(j \geq 1)$ in Y such that $\mathbf{r}_i$ IS-A $\mathbf{y}_j$. Thus, Definition 21 reflects the global nature of the child_of relationship in the sense that for each one of the roots of the area diagram, there must exist parent concepts in each of the parent areas. This further implies that each concept in the area must have ancestor concepts in each parent area since one of the roots of the area is its ancestor.

Figure 5.2 shows an example CV with thirteen concepts. In this example, concepts within a box have the same property set. Consider generating the original area diagram of this CV. Concepts within a box will be in the same area in the original area diagram. Therefore, concepts **E** and **F** will be grouped into one area. By our naming convention, this area is called *E*(2) Area*. Note that the *D Area* has the property set $\{a, b\}$ which is a subset of *C Area*'s property set $\{a, b, c\}$. By Definition 21, *E*(2) Area* is a child_of both *C Area*, *D Area*, and *H Area*. This is because parents of both **E** and **F** are in each of the areas, *C Area*, *D Area*, and *H Area*.

Figure 5.3 shows the original area diagram of Figure 5.2. Areas are represented by boxes labeled by their names. Attributes are listed in the box where they are introduced under a horizontal line. The child_of relationships are represented by

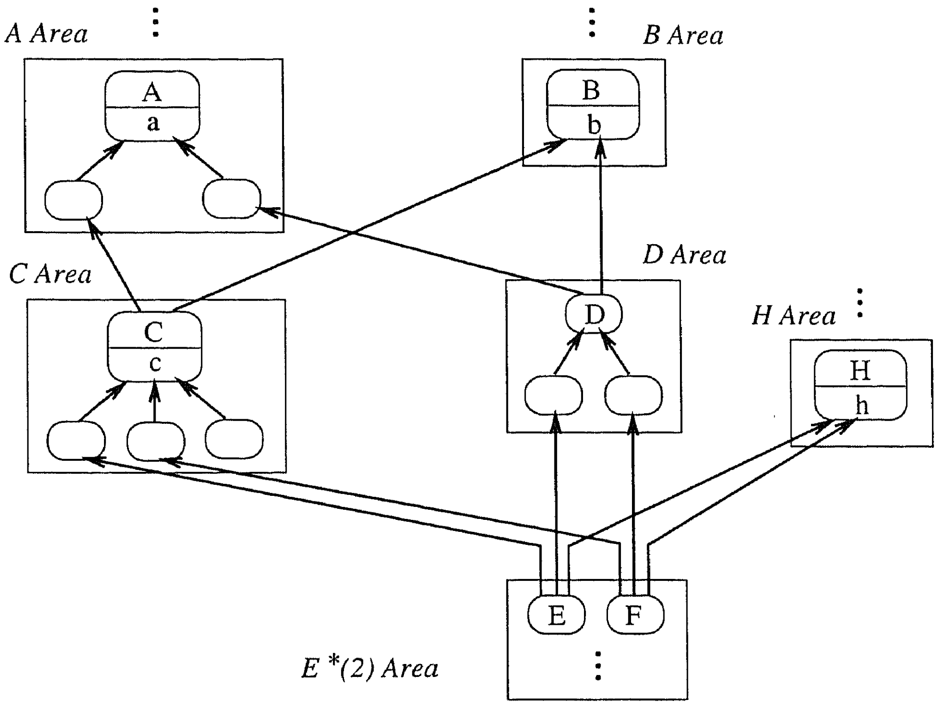


Figure 5.2 Example CV with four property-introducing concepts and three intersection concepts

solid arrows, and carry out the property inheritance in area diagrams. For example, the *E*(2) Area* inherits the property set $\{a,\ b,\ c\}$ from *C Area*, the property set $\{a,\ b\}$ from *D Area*, and the property set $\{h\}$ from *H Area*. By definition, child_of relationships are not transitive.

Similar to the IS-A$'$ relationship defined in Chapter 4, we need to define a relationship which can reflect IS-A relationships of non-roots IS-A relationships of roots. We will reuse the name IS-A$'$ for such a relationship which is defined as follows. The IS-A$'$ relationship complements the child_of relationship of the global nature.

Definition 22: (IS-A$'$) Let X and Y be areas in an area partition where X is not a child_of Y. If there exists a concept **c** in X and a concept **d** in Y such that **c** IS-A **d**, then X IS-A$'$ Y. □

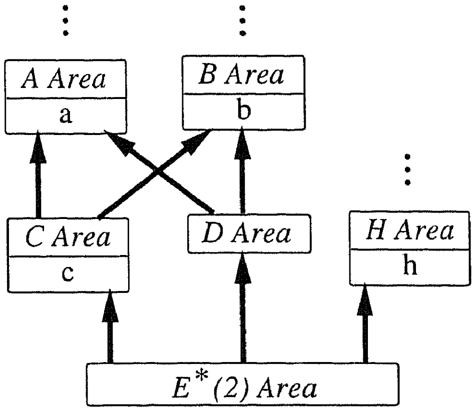


Figure 5.3 Area diagram for CV shown in Figure 5.2

If two areas have an IS-A$'$ relationship between them, we know that some concepts in these two areas are connected by IS-A relationships. Since the child_of relationships allow short-cuts and the IS-A$'$ relationships reflect IS-A links which do not covered by child_of relationships, the problems of short-cuts, missing links, and partition boundaries described in Section 4.1 will not arise. Note that the IS-A$'$ relationship is not transitive, i.e., C IS-A$'$ B and B IS-A$'$ A does not imply C IS-A$'$ A. Similar to the IS-A$'$ relationship in a singly-rooted schema, the IS-A$'$ relationship in an area diagram is also a hierarchical relationship. It is unlike the global child_of relationship which captures the IS-A semantics of all concepts. Instead IS-A$'$ relationships model the fact that some concepts (not all) in a certain area have parents in another area. In other words, let A, B be two areas and B IS-A$'$ A. Then, there must exist some concepts in B which do not have parents in A, otherwise, the relationship would be child_of and not IS-A$'$. With the definitions of child_of and IS-A$'$ relationships, we can define the area diagram as follows. The nature of the IS-A$'$ relationship is demonstrated in Figure 5.4 where the root **G** does not have IS-A relationships to areas *C Area*, *D Area*, and *H Area* like the roots **E** and **F** do. Another possibility is that a non-root concept of an area has an IS-A relationship to a non parent area (see the IS-A relationship from $\mathbf{y}_3$ to $\mathbf{x}_3$ in Figure 5.6 and Figure 5.7).

Definition 23: (Area Diagram [equivalent redefinition]) An area diagram **AD(V)** is a quintuple (**V**, f, **PR**, $\mathcal{C}$, $\mathcal{I}$), where **V** is a CV, f is the function which maps concepts into areas, **PR** is the area partition derived from f, $\mathcal{C}$ is a set of child_of relationships defined between areas in the area partition, $\mathcal{I}$ is a set of IS-A$'$ relationships defined between areas in **PR(V)**. □

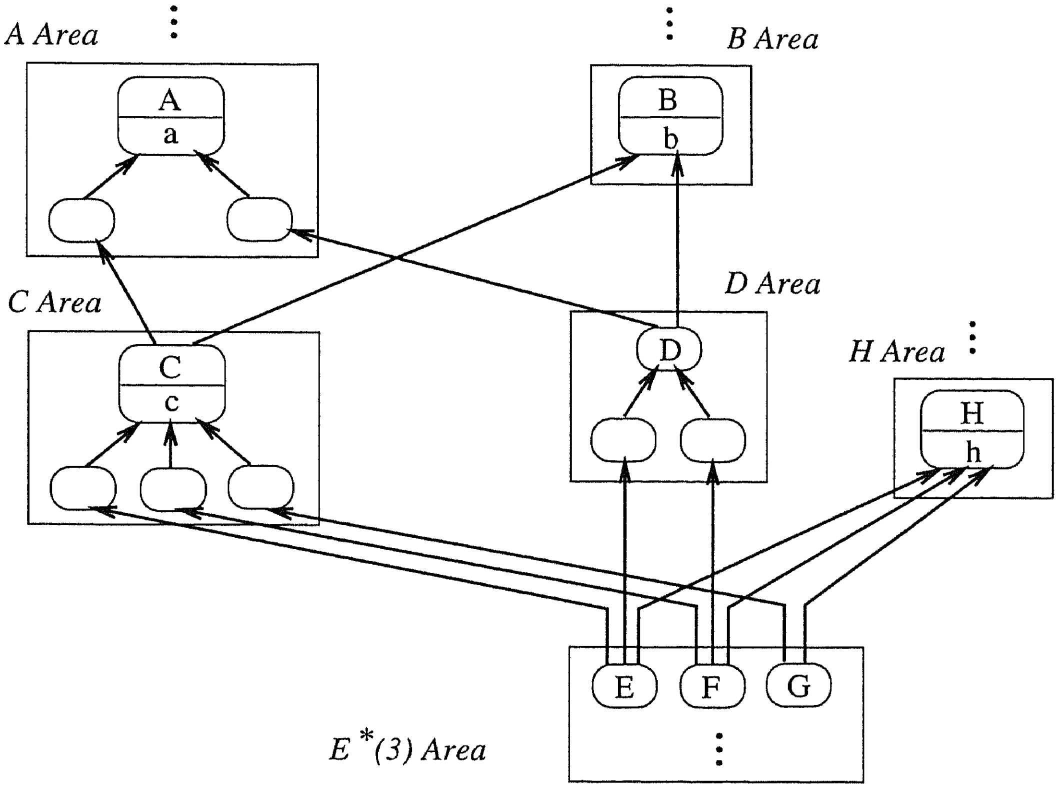


Figure 5.4 Example CV with four property-introducing concepts and four intersection concepts

Figure 5.4 shows an example identical to Figure 5.2 except that there is an extra root **G** in the bottom area. The concepts in the boxes have the same property sets. Let us note that concept **G** inherits the property set {a, b, c} from its parent in *C Area* and inherits the property set $\{h\}$ from its parent in *H Area*. As a result, the concepts **E**, **F**, and **G** have the same property set.

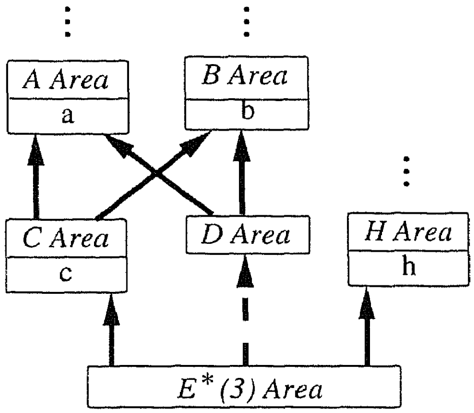


Figure 5.5 Area diagram for CV shown in Figure 5.4

Figure 5.5 shows the original area diagram of Figure 5.4. By our naming convention, the bottom area will be named *E* (3) Area* (see Figure 5.5). Furthermore, the relationship between the *E* (3) Area* and *D Area* is not child_of anymore since one root (**G**) does not have any parent residing in *D Area*. The relationship between these two areas is IS-A$'$ induced from the IS-A relationships of the concepts **E** and **F**. From this area diagram, users know that all concepts in the *E* (3) Area* are descendents of the concepts **C** and **H** (the roots of *C Area* and *H Area*, respectively). This is because, first, the *E* (3) Area* is a child_of *C Area* and *H Area*. That means all roots have parents in *C Area* and *H Area*. Furthermore, the two concepts **C** and **H** are property-introducing concepts. Hence, *C Area* and *H Area* are singly-rooted. Thus, all concepts in *C Area* (*H Area*) are descendents of the concept **C** (**H**). Therefore, concepts in the *E* (3) Area* must be descendants of **C** and **H**. On the other hand, the IS-A$'$ relationship between *E* (3) Area* and *D Area* tells us that not every concept in the *E* (3) Area* is a descendant of the root of *D Area*. Only some concepts in this area have IS-A relationships pointing to concepts in *D Area*, and there must be some concepts in this area which do not have parents in *D Area*.

5.3 The Multilevel Area Diagram Model

As mentioned earlier in this chapter, we want to have a system which can provide more than one abstraction of a CV to enable users to view a CV at different levels of abstraction at their choice. In this section, we provide a method to manage simultaneously multiple area diagrams of a CV by defining an abstraction relation between two area diagrams.

Remember that the singly-rooted area diagram is created from the original area diagram using a finer partition of the multi-rooted intersection areas. Similar to the scenario of switching from the OOVR schema to the concept network, we want to enable a user to switch from the original area diagram to the singly-rooted area diagram, or vice versa. It is necessary to define a *unification relation* between two area diagrams to indicate whether it is possible to switch from one to the other. The process of switching from one area diagram to another area diagram is switch from one abstraction of a CV to another abstraction of the same CV. The switch can be performed from a more "general" area diagram to a more "specific" area diagram, or vice versa. We allow the switch only if the two area diagrams are related by a unification relation.

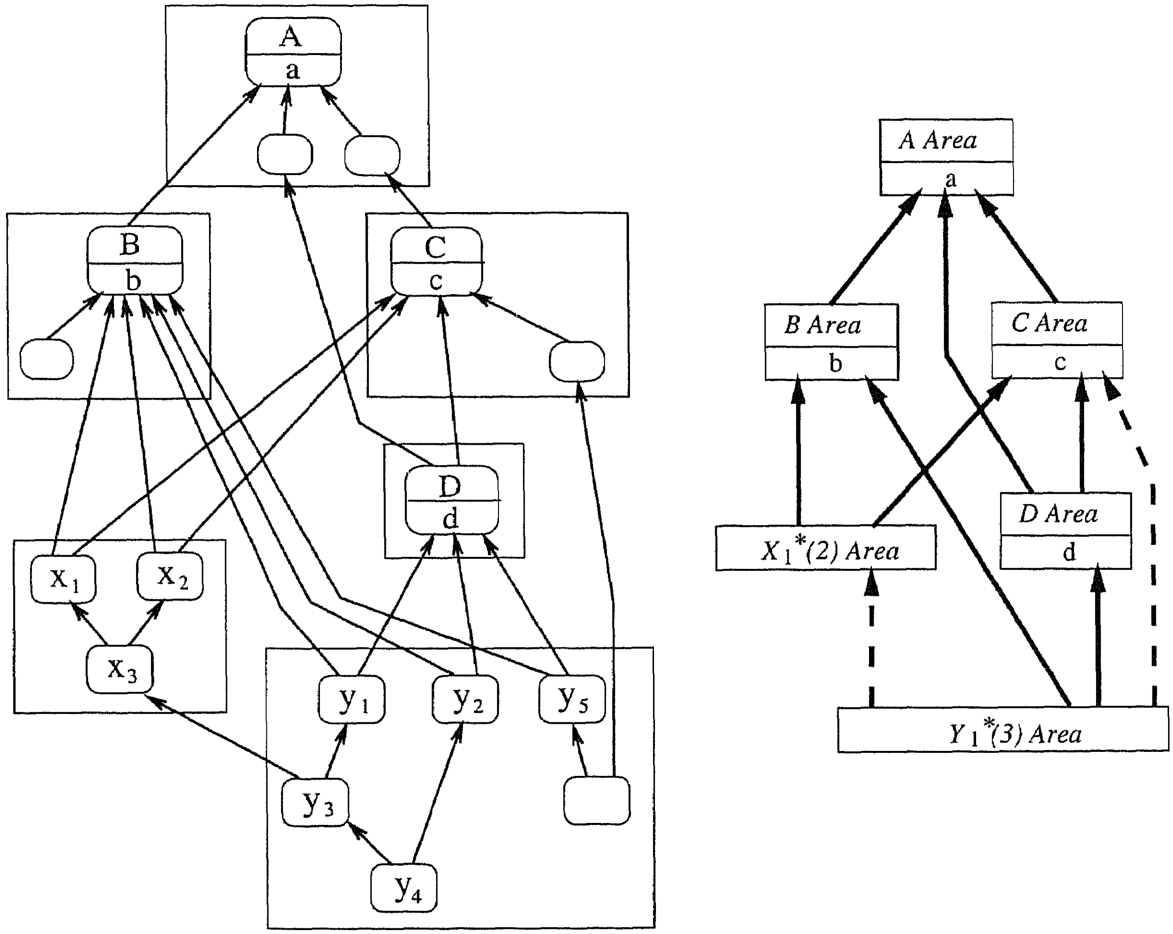
The unification relation between two area diagrams is independent of the relationships *within* these diagrams. We define the unification relation between the two area partitions of the two area diagrams.

Definition 24: (The Unification Relation Between Area Partitions) Let **V** be a CV. Let $\mathbf{PR}_1(\mathbf{V})$ and $\mathbf{PR}_2(\mathbf{V})$ be two different area partitions. The relation $\mathcal{R}^{\mathcal{U}}$: $\mathbf{PR}_1(\mathbf{V}) \rightarrow \mathbf{PR}_2(\mathbf{V})$ is a unification relation if $\forall A \in \mathbf{PR}_1(\mathbf{V})$, the area $\mathcal{R}^{\mathcal{U}}(A) \in \mathbf{PR}_2(\mathbf{V})$ satisfies $A \subseteq \mathcal{R}^{\mathcal{U}}(A)$. □

The condition for a unification relation from $\mathbf{PR}_1(\mathbf{V})$ to $\mathbf{PR}_2(\mathbf{V})$ is that $\mathbf{PR}_1(\mathbf{V})$ is a finer partition than $\mathbf{PR}_2(\mathbf{V})$. Note that ($\mathcal{R}^{\mathcal{U}}$:$\mathbf{AD}_1(\mathbf{V}) \rightarrow \mathbf{AD}_2(\mathbf{V})$)

is equivalent to ($\mathcal{R}^{\mathcal{U}}$:$\mathbf{PR}_1(\mathbf{V}) \rightarrow \mathbf{PR}_2(\mathbf{V})$), where $\mathbf{PR}_1(\mathbf{V})$ and $\mathbf{PR}_2(\mathbf{V})$ are area partitions of the area diagrams $\mathbf{AD}_1(\mathbf{V})$ and $\mathbf{AD}_2(\mathbf{V})$, respectively.

By Definition 24, for any CV **V**, its singly-rooted area diagram $\mathbf{AD}_R(\mathbf{V})$ will have a unification relation to the original area diagram $\mathbf{AD}_S(\mathbf{V})$. Concepts within an area of $\mathbf{AD}_R(\mathbf{V})$ will reside in one area in $\mathbf{AD}_S(\mathbf{V})$. All the areas in $\mathbf{AD}_R(\mathbf{V})$ with the same property set are related to one area of $\mathbf{AD}_S(\mathbf{V})$. On the other hand, no split of an area in the singly-rooted diagram is allowed while switching to the original diagram.

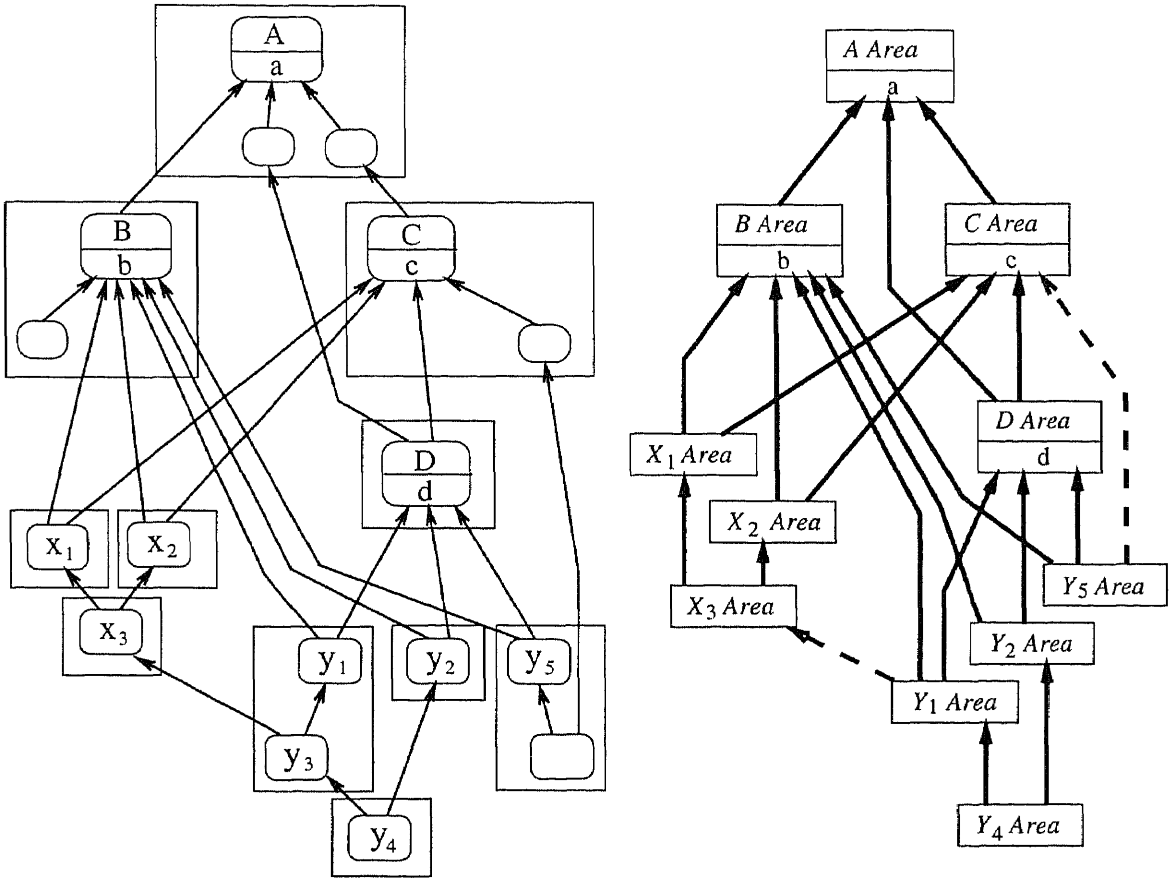


(a)

(b)

Figure 5.6 An example CV (a) and its original area diagram (b)

Figure 5.6 (a) shows an example CV $\mathbf{V}$ and its original area diagram $\mathbf{AD}_S(\mathbf{V})$ in Figure 5.6(b). Boxes in $\mathbf{V}$ will be represented as areas in $\mathbf{AD}_S(\mathbf{V})$. The sixteen concepts in $\mathbf{V}$ are grouped into six areas in the area diagram $\mathbf{AD}_S(\mathbf{V})$, which has eight child_of relationships and two IS-A$'$ relationships.



(a) (b)

Figure 5.7 An example CV (a) and its singly-rooted area diagram (b)

In Figure 5.7 (a), we show the singly-rooted area diagram for the same CV as in Figure 5.6. In Figure 5.7 (b), we show its singly-rooted area diagram $\mathbf{AD}_R(\mathbf{V})$. There are eleven areas in $\mathbf{AD}_R(\mathbf{V})$ with eighteen child_of relationships and two IS-A$'$ relationships. According to Definition 24, $\mathcal{R}^{\mathcal{U}}$:$\mathbf{AD}_R(\mathbf{V})\rightarrow\mathbf{AD}_S(\mathbf{V})$. The Y_1**(3) Area* in $\mathbf{AD}_S(\mathbf{V})$ [see Figure 5.6 (b)] is refined into the following four areas in $\mathbf{AD}_R(\mathbf{V})$: Y_1 *Area*, Y_2 *Area*, Y_4 *Area* and Y_5 *Area* [see Figure 5.7 (b)]. Similarly, the X_1 **(2)*

Area in $\mathbf{AD}_S(\mathbf{V})$ [see Figure 5.6 (b)] is refined into X_1 *Area*, X_2 *Area*, and X_3 *Area* in $\mathbf{AD}_R(\mathbf{V})$ [see Figure 5.7 (b)]. These two area diagrams capture two different levels of abstraction of a CV. By combining them together, we obtain the Multilevel Area Diagram model. To demonstrate the refinement relation between area diagrams in Figure 5.8, the notation $\mathcal{R}^{\mathcal{U}}(M)=N$ denotes that M is an area in the singly-rooted area diagram in Figure 5.8(a) and N is an area in the original area diagram in Figure 5.8(b). Hence, $\mathcal{R}^{\mathcal{U}}(Y_1\ Area)=\mathcal{R}^{\mathcal{U}}(Y_2\ Area)=\mathcal{R}^{\mathcal{F}}(Y_5\ Area)=Y_1\ ^*(3)\ Area$, $\mathcal{R}^{\mathcal{U}}(X_1\ Area)=\mathcal{R}^{\mathcal{U}}(X_2\ Area)=\mathcal{R}^{\mathcal{U}}(X_3 Area)=X_1\ ^*(2)\ Area$, $\mathcal{R}^{\mathcal{U}}(A\ Area)=A\ Area$, $\mathcal{R}^{\mathcal{U}}(B\ Area)=B\ Area$, $\mathcal{R}^{\mathcal{U}}(C\ Area)=C\ Area$, and $\mathcal{R}^{\mathcal{U}}(D\ Area)=D\ Area$.

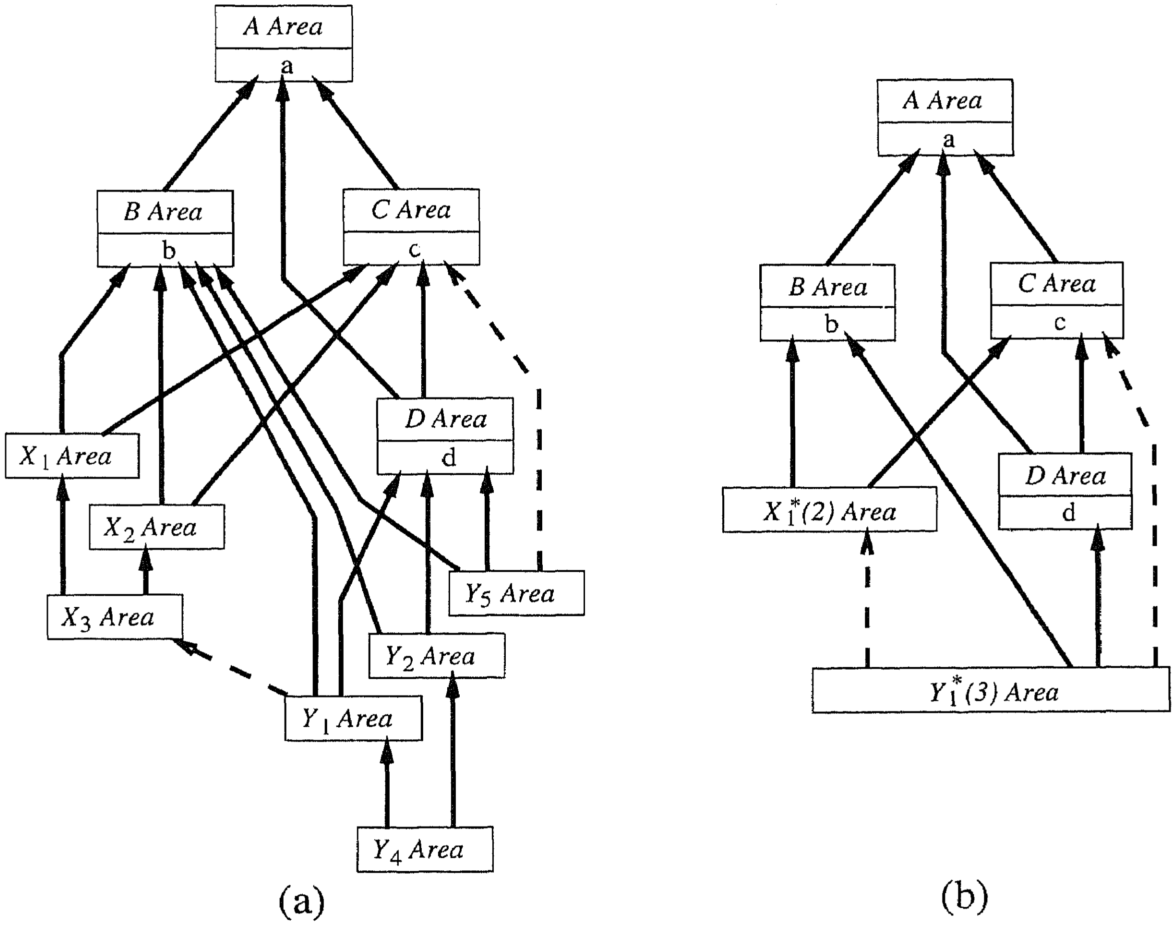


Figure 5.8 The singly-rooted area diagram (a) and (b) the original area diagram of the CV in Figure 5.6(a)

As will be discussed in the next section, the definition of the Multilevel Area Diagram is designed for easy extension to more than two area diagrams.

Definition 25: **(Multilevel Area Diagram [MLAD])** Let $\mathbf{AD}_1$ and $\mathbf{AD}_2$ be two area diagrams generated for a CV with the refinement relation $\mathcal{R}^{\mathcal{U}}$ connecting them. The Multilevel Area Diagram is the quadruple ($\mathbf{V}$, $\mathcal{R}^{\mathcal{U}}$, $\mathbf{AD}_1$, $\mathbf{AD}_2$). □

Given a CV $\mathbf{V}$, consider the original area diagram $\mathbf{AD}_S(\mathbf{V})$ and the singly-rooted area diagram $\mathbf{AD}_R(\mathbf{V})$ obtained following the techniques of Chapter 2 and Chapter 3. Let $\mathcal{R}^{\mathcal{U}}$ be the inverse refinement relation from $\mathbf{AD}_R(\mathbf{V})$ to $\mathbf{AD}_S(\mathbf{V})$. Then ($\mathbf{V}$, $\mathcal{R}^{\mathcal{U}}$, $\mathbf{AD}_S(\mathbf{V})$, $\mathbf{AD}_R(\mathbf{V})$) is a MLAD. As a matter of fact, such a MLAD exists for every vocabulary for which $\mathbf{AD}_S(\mathbf{V}) \neq \mathbf{AD}_R(\mathbf{V})$. In the case where these two area diagrams are equal, no MLAD is needed.

According to Definition 19, an "area partition" serves as a partition for the entire CV. That means a concept must belong to exactly one area in an area diagram. In the MLAD framework, more than one area diagram exists. Hence, a concept in a CV may belong to two areas in different levels. For instance, the concept $\mathbf{y}_4$ in Figure 5.6 (a) is a member of the Y_1 **(3) Area* in Figure 5.6 (b). The same concept $\mathbf{y}_4$ in Figure 5.7 (a) is a member of the Y_4 *Area* in Figure 5.7 (b).

In order to find the proper areas containing a given concept $\mathbf{x}$, we use the function f_S and f_R which are parts of the quadruple notation of the area diagrams $\mathbf{AD}_S$ and $\mathbf{AD}_R$, respectively. The function $f_S(\mathbf{x})$ [$f_R(\mathbf{x})$] will produce the area containing $\mathbf{x}$ in the original (singly-rooted) area diagram $\mathbf{AD}_S$ ($\mathbf{AD}_R$). Using these two functions, we not only obtain the proper area, but also know in which area diagram it is. Note that switching between areas from one level (area diagram) to another is facilitated by the refinement relation which should not be confused with the f function which is internal to the area diagram.

5.4 Browsing Examples for MLADs

Let us use Figure 5.6 and Figure 5.7 to demonstrate switching between area diagrams during the browsing of a MLAD. Suppose we start browsing on the original area diagram in Figure 5.6 (right). When we reach the X_1 **(2) Area*, we may want to see a more detailed view. One possibility is switch to the CV directly. However, according to our experience, a multi-rooted intersection area may have a large number of concepts and a very complicated pattern of IS-A connections. In such a case, instead of switching to the CV, users can switch to the more detailed singly-rooted area diagram shown in Figure 5.7 (right). The previous focus area was the X_1 **(2) Area*. Once users switch to the singly-rooted area diagram, they can select one of the roots X_1 *Area* or X_2 *Area* to be the focus area. Let us assume that the user selects X_2 *Area* and navigates to X_3 *Area* via the child_of relationship and then to Y_1 *Area* through the IS-A$'$ relationship. Since Y_1 *Area* IS-A$'$ (not child_of) X_3 *Area*, the root concept of Y_1 *Area* does not have IS-A connections to any concept in X_3 *Area*. By checking the property sets of X_3 *Area* and Y_1 *Area*, we know that P(X_3 *Area*) $\neq$ P(Y_1 *Area*). This means we have entered a p-area derived from a multi-rooted intersection area which is different from the X_1 **(2) Area*. However, we do not know where Y_1 *Area* is located in that multi-rooted intersection area.

After the multi-rooted intersection area has been partitioned, we will have a hierarchy of p-areas which may contain hundreds of additional p-areas. This multi-rooted intersection area obviously has different semantics from the X_1 **(2) Area*. Users may then want to know the name of that multi-rooted intersection area, since it represents the structure of all concepts in that multi-rooted intersection area. The Y_1 *Area* may be located at the bottom or middle of the area hierarchy which may be too specific for the user's needs. In order to orient themselves within the newly entered multi-rooted intersection area Y_1**(3) Area* which induces Y_1 *Area*, users can switch to the original area diagram. Before we perform the switch, we can put Y_1 *Area* into a buffer for the purpose of switch back later on.

Once users switch to the original area diagram, users can see that Y_1 *Area* in the singly-rooted area diagram is induced by the Y_1 **(3) Area*. It has two parent areas: *B Area* and *D Area*. Note that *D Area* is a sibling of the X_1 **(2) Area*. Once users have achieved an understanding of the newly entered multi-rooted intersection area, they can switch back to the singly-rooted area diagram to continue browsing. The user has now the choice to return to the previously buffered Y_1 *Area* as a focus. Alternatively, he can select one of the roots to be the new focus area. Assume a user reaches Y_4 *Area* from the Y_1 *Area* and decides that this is the area he is looking for; then he can switch to the CV to check the concepts in this area. Let us note that this kind of browsing is advanced over the OOVR framework since there we can have only one OODB schema at a time.

Let us use the MLAD of the MED to demonstrate a browsing example. By the modeling provided in the previous chapters, the MLAD of the MED contains two area diagrams: the original area diagram with 124 areas and the singly-rooted area diagram with 1,835 areas. Assume a user wants to find out the available medicines for treating a nasal inflammation. The browsing will start at the root (*Medical Entity Area*) of the original area diagram. Among the 25 children of *Medical Entity Area*, *Pharmacy Items (drugs and nondrugs) Area* is the obvious one to select among areas with names such as *Order Frequency*, *Specimen*, *Care Plan Protocol*, etc. The *Pharmacy Items (drugs and nondrugs) Area* has two children: *Operating Room Medications Area* and the intersection area "*Anti-Infective Agents* (31) Area*." Since the *Operating Room Medications Area*" specifies the drug treatments for the operating room, and we are looking for a formulary drug, we select the other area to continue our browsing. At this point, the user should switch to the singly-rooted area diagram. This multi-rooted intersection area is partitioned into 42 areas in the singly-rooted area diagram. We will start our browsing by reviewing the 31 root areas in this intersection area. Two of them, *Anti-Inflammatory Agents Area* and *Eye, Ear, Nose, and Throat Preparations Area*, match the information we are looking for. These two areas have one common child, *Eye, Ear, Nose, and Throat Anti-Inflammatory Agents Area*. Once we reach this area, it is fairly clear that we have found the right area. Then we can switch to the CV to find out how many children there are for the root concept, **Eye, Ear, Nose, and Throat Anti-Inflammatory Agents**. From the IS-A connections in the CV, we find 20 children and drugs available for treating nasal inflammation such as **CPMC Drug: Nasonex Nasal 50MCG 17G SP**, **CPMC Drug: Nasalcort Nasal Spray 16.5 GM**, **CPMC Drug: Vancenase AQ Nasal 25 GM**, etc. Those are the drugs we were looking for.

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

In this dissertation, we have addressed the issue of enhancing the comprehensibility and usability of large Controlled Vocabularies. Toward this end, we have developed a number of new representational methodologies for CVs using OODB modeling and technology. A CV is mapped into an equivalent OODB representation called an Object-Oriented Vocabulary Repository (OOVR). The OOVR schema serves not only for storing concepts into an OODB, but also as an important abstraction network. We have also presented a framework called Multilevel Area Diagrams (MLADs) which can manage multiple abstraction views with respect to a given CV. Overall, the contributions of this work can be broken into three major parts: (1) A three-phase OODB modeling technique for mapping a CV into an OOVR representation; (2) A software tool called the OOVR Generator; and (3) A framework called Multilevel Area Diagrams which incorporates two abstraction levels into a single representation.

We have first presented a technology to generate the OOVR representation. Property-introducing concepts and intersection concepts have been identified in a CV, and they respectively root the property-introducing areas and intersection areas. Each area is mapped into an OODB class which represents a grouping of structurally similar concepts. The OOVR schema provides an abstraction view of the concepts' structures. One root of each area is selected as a naming concept to name the area class and define the SUBCLASS_OF relationships between classes.

In the second phase, every multi-rooted intersection area is partitioned into p-areas which are singly-rooted. Every area and p-area of the vocabulary then has exactly one root. All concepts in an area (all concepts in a p-area) have the same properties, making the area structurally uniform. In addition, all concepts in an area (in a p-area) are descendents of the area root, which makes them semantically similar.

In the third phase, a semantic relationship IS-A$'$ is introduced to overcome the browsing difficulties that arise from missing connections between area classes. The IS-A$'$ relationships represent the IS-A connections which cannot be reflected by the SUBCLASS_OF relationships. With this IS-A$'$ relationship, the OOVR schema provides a more refined abstraction network of a CV for users.

To complement our theoretical methodology, we have built a program called the OOVR Generator which automatically carries out the conversion of a semantic network vocabulary to the OOVR representation. We discussed the architectural details of this program. Interestingly, the OOVR Generator is a "second-order" process, with some of its modules being constructed by other modules during execution. Overall, the OOVR takes as its input a CV contained in two text files, called the Property Description File and the Concept Specification File, and produces a completely functioning OOVR. No human intervention in the form of database modeling or populating is required.

To demonstrate our methodology, we have applied the OOVR Generator to two CVs from the medical domain, the mid-sized InterMED and the very large MED. They are currently up and running on top of ONTOS. The InterMED-OOVR is accessible via the Web [31]. While our demonstrations focused on medical CVs, let us emphasize that our methodology is completely general. It works for any semantic network CV in any subject area, as long as it is possible to map the CV into the Property Description File and Concept Specification File formats which we have described, and as long as the "uniqueness of property introduction" rule is satisfied. The OOVR Generator contains a module, called the Preprocessor, which performs any necessary mapping into the required file formats and which has to be rewritten for different semantic network formalisms. Other than that, the OOVR Generator is completely general.

Different from the OODB representation which has only one abstraction view (i.e., its schema), the Multilevel Area Diagrams (MLADs) framework can manage two or more abstraction networks of a CV at the same time. Each layer is represented as an area diagram with child_of and IS-A$'$ hierarchical semantics relationships. To some degree, these two relationships are similar to SUBCLASS_OF and IS-A$'$ relationships in the OODB framework. The singly-rooted area diagram in an MLAD not only represents an abstraction of a CV, but also represents a refinement of the original area diagram. We have provided the definition of the unification relation between these two area diagrams. The MLAD framework provides an environment which allows users to browse on different levels and switch back and forth between the vocabulary and the two kinds of area diagrams. As we demonstrated, this framework provides users with a comprehensive environment in which to browse the CV for the purpose of retrieval and maintenance.

In summary, we have provided two overall approaches for modeling CVs: the OOVR representation and the MLAD framework. These techniques can help future CV designers implement and maintain large vocabularies as well as help users to comprehend the contents of existing CVs. Theoretical characteristics, such as the absence of cycles, of these representations have been identified and proved. We have developed software tools to generate the OOVR on top of a well supported database system.

An interesting issue of future research is to extend the MLAD to model more than two abstraction views. The way the MLAD is defined, it can be extended to model more than two area diagrams. As a matter of fact, in [40], we have presented a technique to partition a CV by similarity. When either of these techniques is applied to the areas and p-areas of the singly-rooted area diagram, they result in a refinement of the singly-rooted area diagram and a unification relation can be defined. The new area diagram can be considered as a third layer of the MLAD. The quadruple of the MLAD ($\mathbf{V}$, $\mathcal{R}^{\mathcal{U}}$, $\mathbf{AD}_1$, $\mathbf{AD}_2$) needs to be modified to model the third or even several additional area diagrams. It is an interesting open problem to thoroughly investigate the MLAD model for more than two area diagrams.

REFERENCES

1. American Medical Association, Chicago, IL. *Physicians' Current Procedural Terminology: CPT. 4th ed.*, 1998.

2. American Society of Hospital Pharmacists, Bethesda, MD. *American Hospital Formulary Service Drug Information*, 1997. Updated annually.

3. E. Bertino and L. Martino. *Object-Oriented Database Systems, Concepts and Architectures.* Addison-Wesley Publishing Co., Inc, Redwood City, CA, 1993.

4. R. J. Brachman. On the epistemological status of semantic networks. In N. V. Findler, editor, *Associative Networks: Representation and Use of Knowledge by Computers*, pages 3–50. Academic Press, Inc., New York, NY, 1979.

5. R. J. Brachman and J. G. Schmoize. An overview of the KL-ONE knowledge representation system. *Cognitive Science*, 9:171–216, 1985.

6. M. Buchheit, F. M. Donini W. Nutt, and A. Schaerf. Terminological system reviewd: Terminology = Schema + Views. In F. Baader, M. Buchheit, M. A. Jeusfeld, and W. Nutt, editors, *Proceedings of 1st Workshop KRDB'94*, Saarbrücken, Germany, September 1994.

7. M. Buchheit, F. M. Donini W. Nutt, and A. Schaerf. A refined architecture for terminological systems: Terminology = Schema + Views. *Artificial Intelligence*, 99:209–260, 1998.

8. K. E. Campbell, S. P. Cohn, C. G. Chute, G. Rennels, and E. H. Shortliffe. Computer-based support for evolution of a convergent medical terminology. In J. J. Cimino, editor, *Proc. 1996 AMIA Annual Fall Symposium*, pages 269–273, Washington, DC, October 1996.

9. M. J. Carey, D. J. DeWitt, J. E. Richardson, and E. J. Shekita. Storage management for objects in EXODUS. In W. Kim and F. H. Lochovsky, editors, *Object-Oriented Concepts, Databases, and Applications*, pages 341–369. ACM Press, New York, NY, 1989.

10. R. G. G. Cattell and D. K. Barry, editors. *The Object Database Standard: ODMG 2.0.* Morgan Kaufmann Publishers, Inc., San Francisco, CA, 1997.

11. J. J. Cimino. Personal communication, January 1997. Department of Medical Informatics, Columbia University.

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

13. J. J. Cimino, G. Hripcsak, S. B. Johnson, and P. D. Clayton. Designing an introspective, multipurpose, controlled medical vocabulary. In *Proc. Thirteenth Annual Symposium on Computer Applications in Medical Care*, pages 513–517, Washington, DC, November 1989.

14. College of American Pathologists, Northfield, IL. *Systematized Nomenclature of Human Medicine*, 1976.

15. Columbia Presbyterian Medical Center, New York, NY. *Columbia Presbyterian Medical Center Medical Entities Dictionary*, 1993.

16. J. E. Conklin. Hypertext: An introduction and survey. *IEEE Computer*, 20(9):17–41, 1987.

17. The CYC Corp. URL: http://www.cyc.com/ (24 Nov. 1996).

18. O. Deux et al. The story of O_2. *IEEE Trans. Knowledge and Data Eng.*, 2(1):91–108, 1990.

19. O. Deux et al. The O_2 system. *Communications of the ACM*, 34(10):34–48, October 1991.

20. S. Even. *Graph Algorithms.* Computer Science Press, Potomac, MD, 1979.

21. B. Falkenhainer, A. Farquhar, D. Bobrow, R. Fikes, K. Forbus, T. Gruber, Y. Iwasaki, and B. Kuipers. CML: A compositional modeling language. Technical Report KSL-94-16, KSL, Stanford University, CA, 1994.

22. B. Falkenhainer and K. D. Forbus. Compositional modeling: Finding the right model for the job. *Artificial Intelligence*, 51(1-3):95–143, 1991.

23. A. Farquhar, R. Fikes, and J. Rice. The Ontolingua Server: a tool for collaborative ontology construction. Technical Report KSL-TR-96-26, KSL, Stanford University, CA, 1996.

24. R. Fielding, J. Gettys, J. Mogul, H. F. Nielsen, and T. Berners-Lee. *HTTP Version 1.1.* UC Irvine, CA, 1997.

25. D. H. Fischer. Consistency rules and triggers for thesauri. *Int. Classif.*, 18(4):212–225, 1991.

26. D. H. Fischer. Consistency rules and triggers for multilingual terminology. In *Proc. TKE'93, Terminology and Knowledge Engineering*, pages 333–342, 1993.

27. Kathleen Fisher and John C. Mitchell. On the relationship between classes, objects, and data abstraction. *TAPOS*, 4(1):3–25, 1998.

28. M. S. Fox, J.F. Chionglo, and F. G. Fadel. A common sense model of the enterprise. In *Proceedings of the 2nd Industrial Engineering Research Conference*, pages 425–429, 1993.

29. M. S. Fox and M. Gruninger. An organisation ontology for enterprise modelling: Preliminary concepts for linking structure and behaviour. *Computers in Industry*, 29:123–134, 1996.

30. M. S. Fox and M. Gruninger. Enterprise modelling. *AI Magazine*, 19:109–121, Fall 1998.

31. J. Geller, M. Halper, and Y. Perl. Hybrid diagram/form interface: A two-layered web-based interface to an OODB vocabulary. Submitted for journal publication, 1999.

32. M. R. Genesereth and R. E. Fikes. Knowledge Interchange Format, version 3.0. Reference Manual Logic-92-1, Computer Science Department, Stanford University, CA, 1992.

33. C. A. Goble, S. K. Bechhofer, W. D. Solomon, A. L. Rector, W. A. Nowlan, and A. J. Glowinski. Conceptual, semantic and information models for medicine. In *Proc. 4th European-Japanese Seminar on Information Modelling and Knowledge Bases*, Kista, Sweden, May 1994.

34. C. A. Goble, P. Crowther, and D. Solomon. A medical terminology server. In D. Karagiannis, editor, *Database and Expert Systems Applications(DEXA-94)*, pages 661–670. Springer, Berlin, Heidelberg, 1994.

35. C. A. Goble, A. J. Glowinski, W. A. Nolan, and A. L. Rector. A descriptive semantic formalism for medicine. In *Proc. 9th ICDE*, pages 624–631, Vienna, Austria, 1993.

36. A. Goldberg and D. Robson. *Smalltalk-80: The Language and its Implementation.* Addison-Wesley Publishing Co., Inc, Reading, MA, 1983.

37. T. R. Gruber. Ontolingua: A mechanism to support portable ontologies. Technical Report KSL-91-66, KSL, Stanford University, CA, 1991.

38. M. Gruninger and M. S. Fox. Methodology for the design and evaluation of ontologies. In *Workshop on Basic Ontological Issues in Knowledge Sharing, IJCAI-1995*, Montreal, CANADA, 1995.

39. H. Gu, J. J. Cimino, M. Halper, J. Geller, and Y. Perl. Utilizing OODB schema modeling for vocabulary management. In J. J. Cimino, editor, *Proc. 1996 AMIA Annual Fall Symposium*, pages 274–278, Washington, DC, October 1996.

40. H. Gu, J. Geller, M. Halper, and L. Liu. Using a similarity measurement to partition a vocabulary of medical concepts. submitted for publication, 1999.

41. H. Gu, M. Halper, J. Geller, and Y. Perl. Benefits of an OODB representation for controlled medical terminologies. 1999. To appear in *Journal of the American Medical Informatics Association.*

42. H. Gu, L. Liu, M. Halper, J. Geller, and Y. Perl. Converting an integrated hospital formulary into an object-oriented database representation. In C.G. Chute, editor, *Proc. 1998 AMIA Annual Fall Symposium*, pages 770–774, Orlando, FL, November 1998.

43. H. Gu, Y. Perl, J. Geller, M. Halper, J. Cimino, and M. Singh. Partitioning a vocabulary's IS-A hierarchy into trees. In D. R. Masys, editor, *Proc. 1997 AMIA Annual Fall Symposium*, pages 630–634, Nashville, TN, October 1997.

44. G. Guerrini, E. Bertino, B. Catania, and J. Garcia-Molina. A formal views of object-oriented database systems. *TAPOS*, 3(3):157–183, 1997.

45. M. Hammer and D. McLeod. Database description with SDM: A semantic database model. *ACM Transactions on Database Systems*, 6(3):351–386, 1981.

46. S. Heiler and S. B. Zdonik. Object views: Extending the vision. In *In IEEE International Conference on Data Engineering*, pages 86–93, 1990.

47. R. Hull and R. King. Semantic database modeling: Survey, applications, and research issues. *ACM Computing Surveys*, 19(3):201–260, September 1987.

48. B. L. Humphreys, D. A. B. Lindberg, H. M. Schoolman, and G. O. Barnett. The Unified Medical Language System: An informatics research collaboration. *Journal of the American Medical Informatics Association*, 5(1):1–11, 1998.

49. N. Ide, J. Le Maitre, and J. Véronis. Outline of a model for lexical databases. *Information Processing and Management*, 29(2):159–186, 1993.

50. Y. Iwasaki, A. Farquhar, R. Fikes, and J. Rice. A web-based compositional modeling system for sharing of physical knowledge. In Martha E. Pollack, editor, *Procof the fifteenth International Joint Conference on Artificial Intelligence*, pages 494–500, Nagoya, Japan, August 1997. Morgan Kaufmann.

51. P. D. Karp, K. Myers, and T. Gruber. The generic frame protocol. In *Proc. IJCAI-95*, pages 768–774, Montreal, Canada, 1995.

52. P. D. Karp and S. M. Paley. Knowledge representation in the large. In *Proc. IJCAI-95*, pages 751–758, Montreal, Canada, 1995.

53. M. Kifer, W. Kim, and Y. Sagiv. Querying object-oriented databases. In *Proc. 1992 ACM SIGMOD Conference on Management of Data*, San Diego, CA, June 1992.

54. W. Kim and F. H. Lochovsky, editors. *Object-Oriented Concepts, Databases, and Applications.* ACM Press, New York, NY, 1989.

55. B. M. Kramer, V. K. Chaudhri, and M. Koubarakis. Implementing Telos. In *ACM SIGART Bulletin*, number 3, pages 77–83, 1991.

56. K. M. Kudla and M. C. Rallins. SNOMED: A controlled vocabulary for computer-based patient records. *Journal of American Health Information Management Association*, 69(6), 1998.

57. H. A. Kuno and E. A. Rundensteiner. Developing an object-oriented view management system. In *Proceedings of IBM Centre for Advanced Studies Conference (CASCON'93)*, Toronto, CANADA, pages 548–562, October 1993.

58. H. A. Kuno and E. A. Rundensteiner. The *MultiView* oodb view system design and implementation. *TAPOS*, 2(3):202–225, 1995.

59. H. A. Kuno and E. A. Rundensteiner. Materialized object-oriented views in *MultiView.* In *Research Issues in Data Engineering Workshop (RIDE-DOM 95)*, pages 78–85, March 1995.

60. H. A. Kuno and E. A. Rundensteiner. Using OO principles tp optimize update propagation to materized views. In *Proc. 12th ICDE*, pages 310–317, 1996.

61. K. Y. Lai, T. W. Malone, and K. C. Yu. Object lens: A "spreadsheet" for cooperative work. *IEEE Trans. Office Information Systems*, 6(4):332–353, 1988.

62. C. Lamb, G. Landis, J. Orenstein, and D. Weinreb. The ObjectStore database system. *Communications of the ACM*, 34(10):50–63, October 1991.

63. F. Lehmann. Semantic networks. In F. Lehmann, editor, *Semantic Networks in Artificial Intelligence*, pages 1–50. Pergamon Press, Tarrytown, NY, 1992.

64. F. Lehmann, editor. *Semantic Networks in Artificial Intelligence.* Pergamon Press, Tarrytown, NY, 1992.

65. D. B. Lenat and R. V. Guha. *Building Large Knowledge-Based Systems: Representation and Inference in the Cyc Project.* Addison-Wesley Publishing Co., Inc., Reading, MA, 1990.

66. D. B. Lenat and R. V. Guha. The evolution of cycl, the cyc representation language. In *ACM SIGART Bulletin*, number 3, pages 84–87, 1991.

67. D. A. B. Lindberg, B. L. Humphreys, and A. T. McCray. The Unified Medical Language System. *Methods of Information in Medicine*, 32:281–291, 1993.

68. L. Liu, M. Halper, J. Geller, and Y. Perl. Controlled vocabularies in OODBs: Modeling issues and implementation. *Distributed and Parallel Databases*, 7(1):37–65, January 1999.

69 L. Liu, M. Halper, H. Gu, J. Geller, and Y. Perl. Modeling a vocabulary in an object-oriented database. In *CIKM-96, Proc. 5rd Int'l Conference on Information and Knowledge Management*, pages 179–188, Rockville, MD, November 1996.

70. M. E. S. Loomis. *Object Databases - The Essentials.* Addison-Wesley Publishing Co., Inc, Redwood City, CA, 1995.

71. T. W. Malone, K. R. Grant, K. Y. Lai, R. Rao, and D. Rosenblitt. Semistructured messages are surprisingly useful for computer-supported coordination. *IEEE Trans. Office Information Systems*, 5(3):115–131, 1987.

72. T. W. Malone, K. R. Grant, F. A. Turbak, S. A. Brobst, and M. D. Cohen. Intelligent information sharing systems. *ACM Commun.*, 30(5):390–402, 1987.

73. E. Mays, C. Apte, J. Griesmer, and J. Kastner. Experience with K-Rep: An object-centered knowledge representation language. In *Proc. IEEE AI Application Conference*, San Diego, CA, March 1988.

74. G. A. Miller. WordNet: A lexical database for English. *Communications of the ACM*, 38(11):39–41, 1995.

75. W. Möhr and L. Rostek. TEDI: An object-oriented terminology editor. In *Proc. TKE'93, Terminology and Knowledge Engineering*, pages 363–374, 1993.

76. T. J. Mowbray and R. Zahavi. *The Essential CORBA - Systems Integration Using Distributed Objects.* John Wiley & Sons, Inc., New York, NY, 1995.

77. J. Mylopoulos. Conceptual modeling and telos. In P. Loucopoulos and R. Zicari, editors, *Conceptual Modeling, Database, and Case: An Integrated View of information Systems Development*, pages 49–68. John Wiley & Sons, Inc., New York, NY, 1992.

78. J. Mylopoulos, A. Borgida, M. Jarke, and M. Koubarakis. Telos: Representing knowledge about information systems. *TOIS*, 8(4):325–362, 1990.

79. J. Mylopoulos, V. Chaudhri, D. Plexousakis, A. Shrufi, and T. Topaloglou. Building knowledge base management systems: A progress report. Technical Report DKBS-TR-94-4, Department of Computer Science, University of Toronto, 1994.

80. National Library of Medicine, Bethesda, MD. *Medical Subject Headings.* Updated annually.

81. The National Drug Code Directory.
URL: http://www.fda.gov/cder/ndc/index.htm (21 Oct. 1997).

82. M. C. Norrie, U. Reimer, P. Lippuner, M. Rys, and H.-J. Schek. Frames, objects and relations: Three semantic levels for knowledge base systems. In *Proc. of 1st Workshop KRDB*, Saarbrcken, Germany, 1994.

83. N. F. Noy and C. D. Hafner. The state of the art in ontology design: A survey and comparative review. *AI Magazine*, 18(3):53–74, Fall 1997.

84. ONTOS, Inc. Lowell, MA. *ONTOS DB 3.1 Reference Manual*, 1995.

85. The OOVR Browser. URL:
http://object.njit.edu:2000/~newoohvr/JBI/INTERMED/index.html
(12 Aug. 1996).

86. K. Radermacher. Abstraction techniques in semantic modelling. In H. Jaakkola et al., editors, *Information Modelling and Knowledge Bases IV.* IOS Press, Amsterdam, 1993.

87. K. Radermacher. An extensible graphical programming environment for semantic modelling. In R. Cooper, editor, *Interfaces to Database Systems*, pages 353–373. Springer-Verlag, London, 1993.

88. A. L. Rector, S. Bechhofer, C. A. Goble, I. Horrocks, W. A. Nowlan, and W. D. Solomon. The GRAIL concept modelling language for medical terminology. *Artifical Intelligence in Medicine*, 9:139–171, 1997.

89. A. L. Rector, W. A. Nowlan, and A. J. Glowinski. Goals for concept representation in the GALEN project. In *SCAMC'93, the 17th annual Symposium on Computer Applications in Medical Care*, pages 414–418, Washington, USA, 1993.

90. A. L. Rector, W. A. Nowlan, and S. Kay. Conceptual knowledge: The core of medical information systems. In K. C. Lun, P. Degoulet, T. E. Pierre, and Rienhoff, editors, *MEDINFO 92, Proc. of the Seventh World Congreee on Medical Informatics*, pages 1420–1426, Geneva, North-Holland, 1992.

91. M. H. Scholl, C. Laasch, C. Rich, H. J. Schek, and M. Trsch. The COCOON object model. Technical Report 211, Dept. of Computer Science, Swiss Federal Institute of Technology, CH-8092 Zurich, Switzerland.

92. V. Soloviev. An overview of three commercial object-oriented database management systems: ONTOS, ObjectStore, and O_2. *SIGMOD Record*, 21(1):93–104, March 1992.

93. J. F. Sowa. Conceptual graphs as a universal knowledge representation. In [64], pages 75–93.

94. J. F. Sowa. *Conceptual Structures, Information Processing in Mind and Machine.* Addison-Wesley Publishing Co., Inc., Reading, MA, 1984.

95. J. F. Sowa. *Principles of Semantic Networks, Explorations in the Representation of Knowledge.* Morgan Kaufmann Publishers, Inc., San Mateo, CA, 1991.

96. J. F. Sowa. Semantic networks. In S. C. Shapiro, editor, *Encyclopedia of Artificial Intelligence, Second Edition.* John Wiley & Sons, Inc., New York, 1992.

97. K. A. Spackman, K. E. Campbell, and R. A. Cote. SNOMED RT: A reference terminology for health care. In D. R. Masys, editor, *Proc. 1997 AMIA Annual Fall Symposium*, pages 640–644, Nashville, TN, October 1997.

98. P. Srinivasan. Thesaurus construction. In B. Frakes W and R. Baezq-Yates, editors, *Information Retrieval: Data Structures & Algorithms*, pages 161–176. Prentice Hall, Englewood Cliffs, NJ, 1992.

99. M. Stonebraker, E. Wong, P. Kreps, and G. Held. Design and implementation of INGRES. *ACM Transactions on Database Systems*, 1(3):189–222, 1976.

100. B. Stroustrup. An overview of C++. *SIGPLAN Notices*, 21(10):7–18, October 1986.

101. B. Stroustrup. *The C++ Programming Language.* Addison-Wesley Publishing Co., Inc., Redwood City, CA, second edition, 1991.

102. The O_2 Team. A technical overview of the O_2 system. In Pericles Loucopoulos and Roberto Zicari, editors, *Conceptual Modeling, Database, and Case: An Integrated View of information Systems Development*, pages 49–68. John Wiley & Sons, Inc., New York, NY, 1992.

103. TOVE Ontologies. URL: http://www.ie.utoronto.ca/EIL/tove/toveont.html (10 Oct. 1997).

104. M. S. Tuttle and S. J. Nelson. The role of the UMLS in 'storing' and 'sharing' across systems. *Int'l J. Bio-Medical Computing*, 34:207–237, 1994.

105. U. S. Department of Health and Human Services, National Institutes of Health, National Library of Medicine. *Unified Medical Language System*, 1996.

106. United States National Center for Health Statistics, Washington, DC. *International Classification of Diseases: Ninth Revision, with Clinical Modifications*, 1980.

107. P. Valduriez, S. Khoshafian, and G. Copeland. Implementation techniques of complex objects. In *Proc. VLDB '86*, pages 101–109, Kyoto, Japan, August 1986.

108. W. A. Woods. What's in a link: Foundations for semantic networks. In R. J. Brachman and H. J. Levesque, editors, *Readings in Knowledge Representation*, pages 218–241. Morgan Kaufmann Publishers, Inc., San Mateo, CA, 1985.

109. The WordNet. URL: http://www.cogsci.princeton.edu/~wn (12 Oct. 1998).

110. S. B. Zdonik and D. Maier. Fundamentals of object-oriented databases. In S. B. Zdonik and D. Maier, editors, *Readings in Object-Oriented Database Systems*, pages 1–32. Morgan Kaufmann Publishers, Inc., San Mateo, CA, 1990.

111. S. B. Zdonik and D. Maier, editors. *Readings in Object-Oriented Database Systems.* Morgan Kaufmann Publishers, Inc., San Mateo, CA, 1990.

112. J. Zhang. Application of OODB and SGML techniques in text database: An electronic dictionary system. *SIGMOD Record*, 24(1):3–8, March 1995.