COMPUTERS AND BIOMEDICAL RESEARCH 27, 472–507 (1994)

Designing a Controlled Medical Vocabulary Server: The VOSER Project

ROBERTO A. ROCHA,* STANLEY M. HUFF,*,† PETER J. HAUG,*,† AND HOMER R. WARNER*

*Department of Medical Informatics, University of Utah College of Medicine, Salt Lake City, Utah; and †Medical Informatics, Intermountain Health Care, Salt Lake City, Utah

Received September 12, 1994

The authors describe their experience designing a controlled medical vocabulary server created to support the exchange of patient data and medical decision logic. The first section introduces practical and theoretical premises that guided the design of the vocabulary server. The second section describes a series of structures needed to implement the proposed server, emphasizing their conformance to the design premises. The third section introduces potential applications that provide services to end users and also a group of tools necessary for maintaining the server corpus. In the fourth section, the authors propose an implementation strategy based on a common framework and on the participation of groups from different health-related domains. © 1994 Academic Press, Inc.

INTRODUCTION

A key component of any clinical information system is its data dictionary (*1*). Well-designed data dictionaries are responsible for much of the success that current information systems have in dealing with the complexity of medical language. However, the ever increasing need to integrate information coming from different sources, and many times available in incompatible formats, has forced a reevaluation of the data dictionary design requirements and assumptions (*1–3*). The new data dictionary design requires the development of formal representation schemes that are coupled to standardized medical vocabularies. These new data dictionaries implement the open systems architecture model, enabling the development of modular and reusable components that offer a standard interface to several client applications (*4*). The desired functionality of these components includes support for typical data collection and review functions, as well as for medical language processing, data gathering for medical reserach, and data manipulation for decision logic and audit processing.

Recently, a position paper from the Board of Directors of the American Medical Informatics Association (AMIA) emphasized that standards for the

0010-4809/94 $5.00

structure and content of medical record systems are essential in creating an efficient computer-stored medical record (*5*). The AMIA Board of Directors recognizes that given the diversity of the medical information, the "standardized" terminology will likely be a combination of multiple coding systems, and that it will require a common "representation language" in order to provide the desired integration. Similar arguments are found in another recent position paper describing the activities and principles of the Canon Group (*6*). The Canon group highlights the need for a "medical-concept-representation language" (MCRL) as an essential component for data exchange and applications development.

Addressing the problems presented above, we have been designing a controlled medical vocabulary server (VOSER) (*7*). The overall objective of the VOSER project is to create a comprehensive repository of medical concepts that enables the *exchange of patient data and medical decision logic* and that supports the *development and implementation of a wide range of clinical information systems*. VOSER has three logical components: a "vocabulary" component that enables the representation of medical concepts with their lexical and linguistic characteristics, a "data structure" component that establishes a "grammar" of how these medical concepts can be combined to represent medical events, and a "maintenance" component that keeps track of the modifications made to the content or the structure of the server.

The VOSER project is a result of our group's involvement in the area of medical concept representation (*8*), including our participation as subcontractors of the Unified Medical Language System (UMLS) project (*9*), and our collaboration as members of the Canon Group (*6*). In addition, published work of many other groups (*10–21*) has influenced our ideas as well.

VOSER's underlying conceptual model is called Event Definition (ED) (*1*). When fully implemented, the ED model provides the basic structural design for all three components of VOSER. The ED model is based on a conceptual view that a patient's medical record is a sequence of clinical events. The ED model is a template based model that resembles case frames (*22, 23*). An ED template captures clinical data by assigning semantic meaning to the attributes (slots) in the template (Fig. 1). The semantic meaning is obtained by assigning to each attribute an exclusive set of concepts that characterizes its domain. For example, a slot that represents parts of the human body has in its domain concepts like lung, arm, and heart. ED templates are always date and time tagged, supporting the chronological sequence of the clinical events.

Research exploring the ED model has provided important insights into how to approach the complexity of medical language and how to develop computational models that can represent medical information (*24–27*). In addition, an important contribution to VOSER's overall architecture is the experience we accumulated with the development and implementation of medical information systems, such as the HELP Hospital Information System (*28–30*) and the Iliad Expert System (*31*). In particular, the maintenance of PTXT (HELP's data

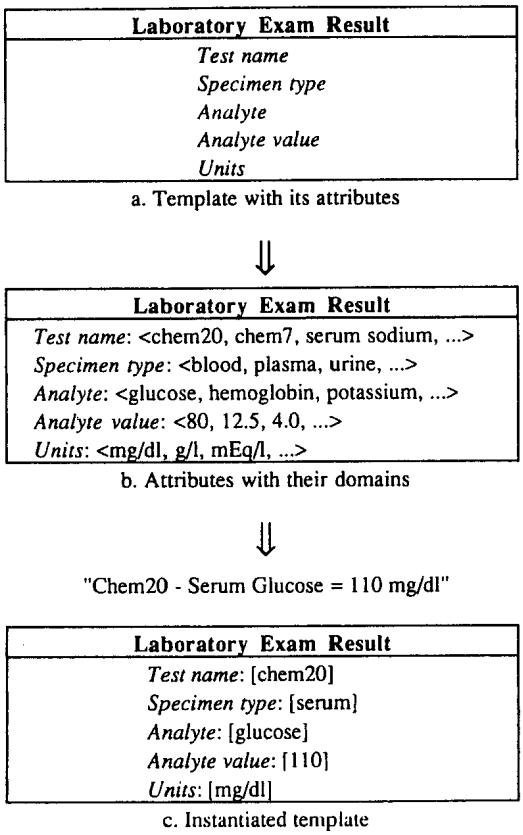

FIG. 1. Overview of the ED model.

dictionary) in different health care facilities has resulted in many critical operational considerations.

More recently, our group has been involved in the development of the Enterprise Lifetime Data Repository (ELDR), a joint initiative of Intermountain Health Care (IHC) and 3M Health Care Systems, and the development of the Ambulatory Care Information System (ACIS), an initiative of the University of Utah Health Sciences Center. These two projects make extensive use of the ED model paradigm, and their data dictionaries are compatible with the VOSER design. ELDR and ACIS are the first two clinical information systems being developed at our site that try to share a common underlying data model. The presence of legacy systems in use by both development sites is creating an even more challenging environment, adding to the VOSER architecture the need to keep the new systems compatible with those already installed.

The present article describes what we have learned by designing an operational vocabulary server. Section I presents important premises that guide the project from both theoretical and practical perspectives. Section II describes the structures that implement the VOSER design, with references to the functional aspects of the ED model. Section III presents the applications necessary to maintain and access VOSER, including client and administrator tools. Finally,

Operational
• Available on-line with real time maintenance
• Support site-specific concepts and terms
• Support site-specific models and configurations
• Enable interchange of medical data
• Enable interchange of medical decision logic

FIG. 2. Summary of the operational premises considered.

Section IV describes the implementation strategy used during a pilot experiment that involved several collaborating groups.

I. PROJECT PREMISES

A central assumption of this project is that before the exchange of medical data is possible among clinical information systems, these systems must agree on the *content* and the *structure* of this data (*5, 6*). In other words, these systems need to converge to a common vocabulary component and to a common data structure component. In the VOSER project, this assumption is fulfilled by using the ED model as the common underlying representation model, i.e., the ED model is used as a MCRL.

In addition to this pivotal assumption, several other assumptions and requirements were considered during the design of VOSER, guiding its logical design and implementation. These premises are here subdivided into two groups: operational premises and technical premises. Operational premises correspond to practical considerations that define what services VOSER should provide and how they should be provided. Technical premises reflect theoretical requirements that enable the implementation of these services. In the following two subsections, those assumptions and requirements that had significant influence over the VOSER design will be presented. Please refer to Figs. 2 and 3 for a complete list of the premises considered.

I.1. Operational Premises

The most important operational consideration is the VOSER overall architecture, i.e., how it will be implemented and accessed. Our initial idea was to establish a true client-server architecture, with a single centralized VOSER providing on-line and real-time services to all its clients (Fig. 4a). With this centralized architecture, all services and communications must be very reliable, given that real-time access will be made during program execution. In addition, VOSER has to have all the vocabulary used by all its clients, including site-specific terms describing names of practitioners, names of departments, administrative items, etc. In this scenario, clients expect minimal turnover time between the submission of a new term and its addition to the server dictionary.

Technical

- Logical model design completely independent of its physical implementation
- Medical concepts and their relationships formally represented using an underlying Information Model
- Fulfill the Vocabulary Model requirements:
  - Unique representation for each concept
    - synonyms, variants, eponyms, polysemyns
  - Support relationships between primitive and compound elements
    - aggregation, decomposition
    - minimal number of primitive elements
    - different granularities
  - Support explicit relationships between elements
    - hierarchical, non-hierarchical
  - Semantically typed
  - Multilingual
  - Enable unambiguous representation of simple and compound concepts - canonical

FIG. 3. Summary of the technical premises considered.

This centralized approach is unlikely to be successful since it violates the autonomy of the client sites, interfering with their operations and retarding their local development. On the other hand, the adoption of this architecture forces integration through the use of a unique common vocabulary and data structure. The biggest advantage is the need for relatively simple maintenance components that control only the centralized server operations.

Considering the unworkable problems described above, the current proposed architecture transforms VOSER into a central repository of only those medical concepts shared by the clients. Client sites refer to this master VOSER when they plan to exchange medical data or medical knowledge with any other client site. This design does not interfere with the client's operational activities, and it does not require real-time services. However, this "freedom" to operate with local models independent of the master server does not help global integration, but, on the contrary, it may cause disarray. We solve this problem by utilizing the ED model as the "standard" interface between local data models and the central repository, i.e., forcing client sites to implement site-specific VOSERs in order to interface with the master VOSER (Fig. 4b). In this scenario, clients have the option of making the client VOSER operational in their systems at runtime, adopting the VOSER model as their internal formalism. Clients also have the option of continuing to utilize their original data models, accessing their client VOSER copies just as an interface to the master VOSER, or to other client VOSERs.

This decentralized approach enables client sites to bypass the server when

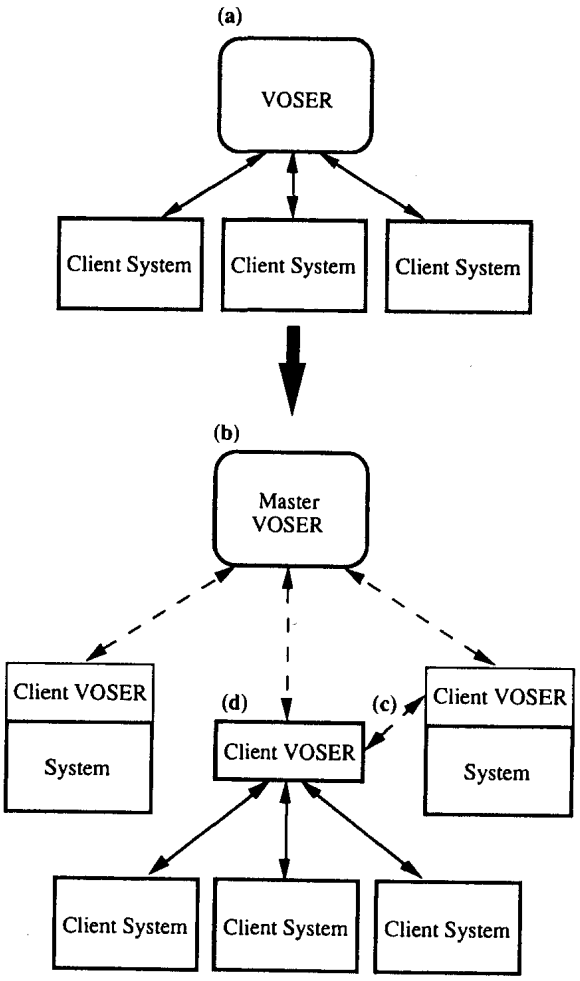

FIG. 4. VOSER proposed architectures: (a) Centralized, (b) Distributed, (c) Client-to-client interface, (d) Regional/Vendor Client.

direct interfaces to other sites become important for performance reasons (Fig. 4c). This design also permits client sites to control "regional" VOSERs that can be shared by facilities located in the same region, or allow vendors to have control over the copies that are distributed to their own customers (Fig. 4d). The decentralization transforms each VOSER into a building block of a hierarchy of VOSERs; i.e., a local VOSER becomes a client of a regional VOSER, that becomes a client of a national VOSER, that may eventually be a client of a multinational VOSER.

In terms of disadvantages, the proposed decentralized architecture increases the complexity of VOSER's maintenance components, consistent with its operation in a distributed environment. It requires VOSER's logical design to be completely independent from its physical implementation, enabling client sites to install VOSER without changing their current hardware and software platforms.

Concept	*Surface forms*
"Parasitic infection caused by the Trypanosoma cruzi"	• South American trypanosomiasis • Chagas' Disease • Chagas-Cruz' Disease • Cruz trypanosomiasis • American trypanosomiasis

Disease template*	
Attribute	*Value*
pathologic process	infection
pathologic process type	parasitic
etiologic agent	Trypanosoma cruzi

*partial display

FIG. 5. The concept "Parasitic infection caused by the *Trypanosoma cruzi*" and some of its surface forms (top). Canonical representation of the concept using the Disease template (bottom).

I.2. Technical Premises

The technical premises considered are extensively documented in the literature (*6, 10, 32–35*). Here we will emphasize only those technical premises that are fundamental to the success of the VOSER project. Please refer to Fig. 3 for a complete list of the technical premises.

A fundamental technical premise is the canonical representation of the medical concepts stored in the server, i.e., for each medical concept identified, there must be one and only one unambiguous form of representing it. Canonical representation is not achieved by listing all possible terms (surface forms) that express a given concept, but by identifying the "smaller" concepts that, when combined, convey, without ambiguity, the meaning of the original concept (*6*). This premise does not interfere with the existence of many surface forms for each concept, and it does not preclude the use of simple ("atomic") concepts or complex ("molecular") concepts by different clinical systems.

Figure 5 shows an example of the concept "Parasitic infection caused by the *Trypanosoma cruzi,*" some of its surface forms, and its canonical storage form. The premise of canonical representation of concepts requires an underlying semantic structure, in our case provided by the ED templates (Fig. 5). The attributes present in these templates guide the identification of the "smaller" component concepts, and, in addition, provide the context that helps to identify nuances of meaning (Fig. 6) (*36*). The names of the attributes from a given template have no special significance except as identifiers of collections of concepts. Therefore, all polysemic forms are distinguished by their participation in different contexts, i.e., meaning is established only in the specific context of use (Fig. 7).

Another important technical premise is the need to represent concepts with

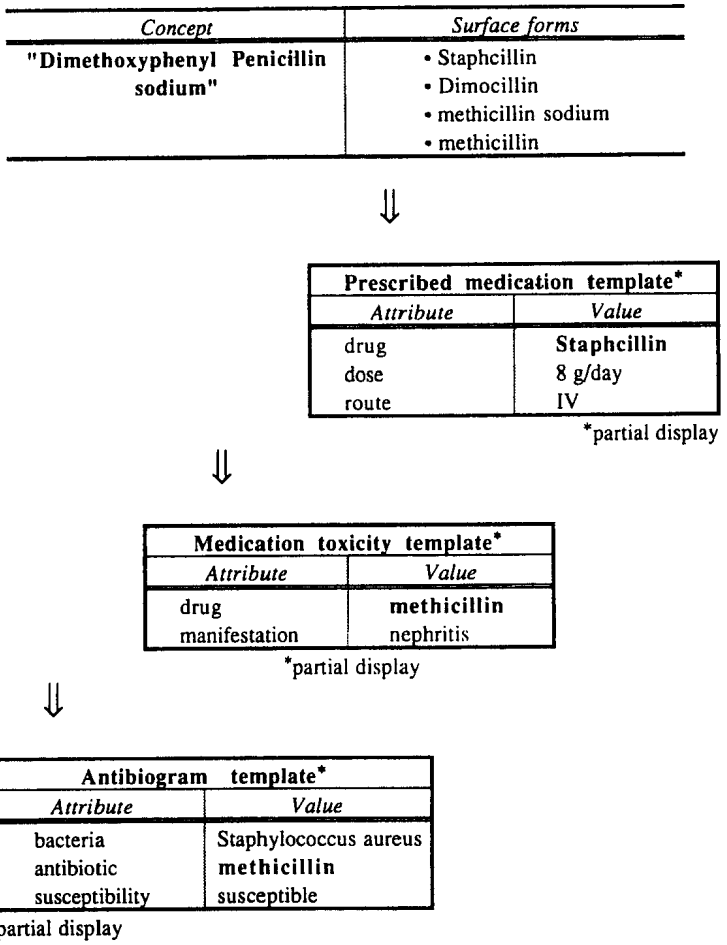

FIG. 6. Interaction between templates and the contextual meaning of concepts.

different granularities, i.e., represent both "atomic" and "molecular" concepts. This premise requires methods on how to aggregate atomic concepts to obtain sensible molecular concepts and how to decompose molecular concepts into their atomic constituents. The ED model again provides the structure that enables VOSER to handle this requirement (Fig. 8). The decomposed form of a given molecular concept may or may not correspond to its canonical storage form (see subsection II.3). We recognize that the storage of atomic concepts is the most flexible option, preventing the unnecessary combinatorial growth of the vocabulary (*37*). However, molecular concepts are represented if any of the client sites utilize them in decision logic or statistical aggregation, or if they are present in one of the external coding schemes, e.g., ICD-9 (*38*), CPT-4 (*39*), etc.

The necessity of supporting different kinds of logical arrangements of elements (taxonomies, meronomies, semantic networks, etc.) is another important technical premise. These arrangements are used to represent many types of relations among concepts, including hierarchical and nonhierarchical relations. Some of these arrangements reflect the structural organization of the vocabula-

Concept	*Surface forms*
"An excretion or substance evacuated."	• discharge[1]
"A setting free, or liberation."	• discharge[2]
"Matter or force set free."	• discharge[3]

⇓

Context	*Example*
clinical finding	vaginal **discharge[1]**.
hospital operation	hospital **discharge[2]**.
etiologic agent	convective **discharge[3]**.

⇓

Template/Cluster	*Attribute*	*Value*
gynecologic finding*	finding	**discharge[1]**
	body region	vaginal
hospital operation*	operation	**discharge[2]**
etiologic agent*	etiologic agent	**discharge[3]**
	agent type	convective
etiologic agent*	etiologic agent	**discharge[3]**
	agent type	disruptive

*partial display

FIG. 7. Polysemic forms distinguished by their existence in different contexts.

Terminology	*Surface form*	*Granularity*
Iliad	Chest X-ray shows cardiomegaly 20.4.18.2.0.0	molecular
Metathesaurus	Chest C0039992.L0008030.S0024330	atomic
Metathesaurus	X-ray C0043309.L0043309.S0001212	atomic
Metathesaurus	exhibits T145	atomic
Metathesaurus	Cardiomegaly C0018800.L0007190.S0022436	molecular

⇓

Template/Cluster	*Attribute*	*Value*
procedure*	procedure	X-ray
procedure*	procedure location	chest
link*	semantic link	exhibits (shows)
observation*	observation	Cardiomegaly

*partial display

FIG. 8. Representation of concepts with different granularities. Decomposition (mapping) of an Iliad term into corresponding Metathesaurus terms.

Concept
"Parasitic infection caused by the Trypanosoma cruzi"

⇓

Language	*Surface forms*
English	• South American trypanosomiasis • Chagas' Disease • Chagas-Cruz' Disease • Cruz trypanosomiasis • American trypanosomiasis
Portuguese	• Tripanossomíase Sul-Americana • Doença de Chagas • Doença de Chagas-Cruz • Tripanossomíase de Cruz • Tripanossomíase Americana • Tripanossomose Sul-Americana
Metathesaurus (version 1.4)	• C0041234.L0007931 • C0041234.L0041229 • C0041234.L0041234 • C0041234.L0176290 • C0041234.L0204883 • C0041234.L0204884 • C0041234.L0204885
SNOMED	• DE-55020
ICD-9	• 086.2

FIG. 9. Representation of multiple languages: terms in different languages, as well as codes from different coding systems, are considered surface forms that represent the same underlying concept.

ries being represented in the VOSER corpus, while others are created to support the implementation of the ED model (see subsection II.4). In addition, these arrangements provide valuable knowledge that can be used by processes designed to maintain the VOSER vocabulary, including computerized matching of concepts, synonym discovery, relationship discovery, etc. (*40, 41*).

An additional technical premise that we judge important is the need to support multiple natural languages. The importance of this premise is well documented in the literature (*13, 42, 43*), and it is already implemented in several terminologies and coding schemes (such as SNOMED (*44*), ICD-9 (*38*), and UMLS (*9*)). Since the primary elements of the VOSER vocabulary are concepts and not their surface forms, a given natural language becomes just another surface form, in other words, just another form of expressing the abstraction (Fig. 9) (see subsection II.1). In fact, all coded identifiers from any coding scheme are also handled in the same way (Fig. 9).

We also recognize that alternative forms of expressing concepts should also be supported. For instance, a given concept may be expressed by graphics, such as icons, bar codes, pictures, wave-form traces, etc.; or by sounds, such as rhythms, noises, etc.; or even by video clips. In all cases, the same paradigm is used, i.e., nontextual objects are treated as additional surface forms.

II. Logical Design

After presenting the important requirements and assumptions that guided this project, we can now introduce the VOSER logical design. As other authors have already concluded, encoding medical language is a complex problem and it requires a rather complex representation model (*6, 37*). The VOSER project logical design is a thorough implementation of the most current ED model structures. In some instances, the ED model structures were modified to allow their use in a distributed environment, like the one created by the proposed VOSER architecture.

The structures presented next apply to both master and client copies of VOSER. The descriptions are kept as simple as possible, limiting the distinction between these two scenarios to cases where it is mandatory to clarify the structural differences.

The current ED model architecture considers each of its structural components a template. Because of this recursive characteristic, the ED model provides the infrastructure to implement itself. In order to prevent terminological problems, the metadata components created to support the ED model implementation are called **VOSER structures,** while the metadata components created to represent the various medical domains are simply called **templates.** The VOSER structures can be subdivided into six functional components:

1. **Master Object Index structures,** where the medical concepts are represented along with their surface forms and some basic lexical details;
2. **Event Definition structures,** where the templates describing the medical events are represented;
3. **Mapping structures,** where the relationships between atomic and molecular concepts are established, including the canonical representation forms;
4. **Configuration structures,** where the necessary types of arrangements of elements are created and maintained;
5. **Maintenance structures,** where the status of the objects and the operations performed are logged and classified;
6. **Application Support structures,** where client-specific features and indexes are supported.

In the following six sections we will briefly introduce these components, giving special attention to features that are particular to the VOSER implementation.

II.1. Master Object Index Structures

The Master Object Index structures can be considered the dictionary and thesaurus components of VOSER. They include the Master Object Index (MOI) itself, plus a collection of auxiliary structures designed to allow the storage of textual definitions, usage examples and annotations, syntactic tags, context specific preferred forms, and a wide range of indexes designed to improve retrieval performance. These indexes range from simple word indexes and keyword indexes to special ones designed to handle each coding scheme, each

Master Object Index (partial display)

*Facility**	*Object ID*	*Concept ID*	*Class**	*Language**	*Name*
18	102	2	4	58	template
18	103	3	4	58	attribute
18	104	4	4	58	vocabulary
18	118	18	4	58	Master VOSER
18	234	58	4	58	English
18	331	64	4	58	Portuguese
18	765	78	4	58	SNOMED
18	876	79	4	58	Metathesaurus v1.4
18	915	80	4	58	PTXT
18	78995	45551	4	64	Hospital de Clínicas
18	6751	1233	4	58	LDS Hospital
18	8541	1431	2	58	Prescribed medication
18	29705	7865	3	58	dose
18	32909	1881	4	58	dyspnea
18	32910	1881	4	58	shortness of breath
18	3289	1881	4	78	F-20040
18	4542	1881	4	79	C0013404.L0013404
1233	8502	1881	4	80	20.1.158.1.101.0.0.0
45551	87660	1881	4	64	falta de ar
45551	87661	1881	4	64	dispnéia

*coded attributes - to decode find the same number in the Concept ID column.

FIG. 10. Partial view of the Master Object Index, with its most important attributes. Some of these attributes (i.e., facility, class, and language) contain codes that are concept IDs of other objects. Objects from different classes and expressed in multiple languages are exemplified.

natural language, and also phonetic and substring searches. In the next paragraphs we will focus only on the description of the MOI.

The MOI is one of the most important elements of the vocabulary server. In the MOI, each entry is considered an instance of a conceptual object. The instances, here called "object instances," are considered objects as well. It is in the MOI that all these conceptual objects are subdivided into classes that describe their role in the system (Fig. 10). Object instances inherit the class from their parent conceptual object. Each object instance is uniquely identified by a combination of two numeric codes, one that identifies the instance itself (*object ID*) and another that identifies the client facility that created the instance (*facility*) (Fig. 10). In addition, each object instance also receives a concept numeric identifier (*concept ID*) (Fig. 10). A given concept ID is shared by all object instances that represent the same conceptual object. For example, "dyspnea" and "shortness of breath," both members of the class of vocabulary objects, share the same concept ID because they are instances of the same conceptual object (Fig. 10). Vocabulary conceptual objects are called *vocabulary concepts,* and their instances are known as *surface forms.*

The abstraction provided by the concept ID enables the clear distinction between generic attributes applicable to the conceptual level, and more specific attributes applicable only at the instance level. For example, definitions and semantic relationships are typical attributes of vocabulary concepts, while capi-

talization, word order, and language, are typical attributes of their surface forms. It is also true that properties of a given conceptual object are always inherited by all its instances.

The VOSER identifiers described in the previous paragraphs, namely object ID, facility, and concept ID, are the internal codes referenced by all the other structures of VOSER. The MOI is the only place where these codes can be decoded. This centralized assignment of codes makes the MOI a self-referenced structure, i.e., codes that apply to the MOI are decoded using the MOI (Fig. 10).

The proposed VOSER architecture designates the MOI as the central element for integrating the vocabulary content. For instance, whenever the addition of a new client vocabulary becomes necessary, the mapping routines try to find the correspondences between the concepts used by this new client vocabulary and the actual concept IDs maintained by the master VOSER. Once the mapping is completed, the client vocabulary is incorporated into the master VOSER, and, if necessary, new concept IDs are created to represent all client vocabulary concepts that were not previously known to the master VOSER. This strategy reduces the translation process to a single step; i.e., instead of mapping each new client vocabulary to all vocabularies represented in the master VOSER, we simply map the new client vocabulary to a set of the master VOSER's concept IDs. The client facility can use this set of concept IDs to establish translations between its own vocabulary and any other vocabulary already available inside the master VOSER (see Fig. 10, where the concept ID = 1881).

The master VOSER concept IDs are called "*master concept identifiers*" (MCIs), and they are considered to be the tokens that express the master VOSER's internal language. MCIs must be assigned and maintained by the master VOSER if we intend to have a fully integrated environment. However, as we mentioned before, the client VOSERs require the freedom to create and maintain their own local vocabulary, keeping the master VOSER from interfering with their local development processes (see subsection I.1). Because of this operational "freedom," client VOSERs can have as many surface forms as they need, including local keywords, mnemonics, abbreviations, etc. Whenever they conclude that a particular object instance should be available in the master VOSER, they can submit it to the review committee that controls the master copy (see Section IV). Once this new object instance is reviewed and sanctioned, it is added to the master VOSER using a combination of the client's facility, the client's object ID, and the corresponding MCI. This strategy makes that particular object instance unique, since each client always has a unique facility identifier.

Another scenario, where a client VOSER proposes the inclusion of a concept that is not known to the master VOSER, is more complicated. Remember that the concept identifier maintained by the master VOSER is considered the MCI, and it is shared by all client VOSERs. Our strategy to enable the clients to create their local concepts, and to eventually incorporate some of these local concepts into the master VOSER, is to utilize different ranges of concept

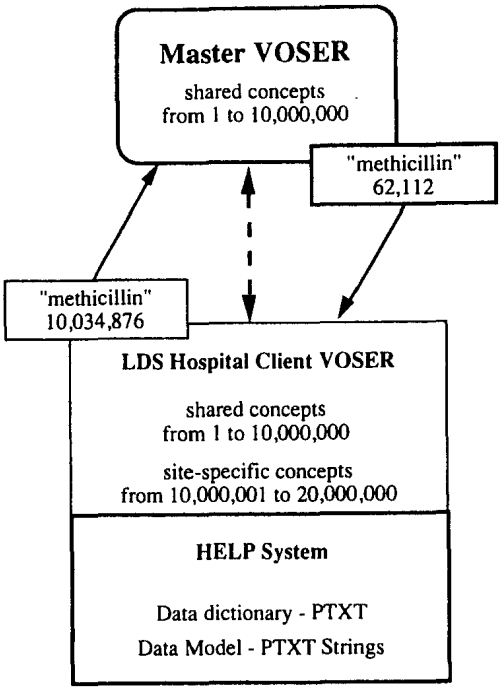

FIG. 11. Exchange of concepts between client and server, using a nonoverlapping range of concept identifiers.

identifiers. The master VOSER MCIs are obtained from a distinct (nonoverlapping) numeric range, lets say from 1 to 10,000,000, while client VOSERs draw their local concept identifiers from 10,000,001 to 20,000,000. For instance, client concepts that are always used locally have their identifiers drawn from the client range. In this case, all clients can use the same range, since these codes have only local meaning. If one of these concepts is eventually added to the master VOSER as a new concept, a new MCI will be required. This new MCI in effect replaces the original client concept ID, and the client VOSER in question becomes responsible for maintaining its local corpus to reflect this change. It should be noted that the same processes can be scaled up to reflect a hierarchy of VOSERs, where a local master VOSER becomes a client of a regional master VOSER. Figure 11 illustrates what we have just described.

In addition to the identification of conceptual objects and instances of these objects, the MOI is also responsible for the morphological classification of the vocabulary objects. This classification is used to define the granularity of each surface form in terms of the number of words used and the number of underlying concepts represented, as follows:

a. **Free morphemes** are single words that represent single concepts, such as "heart," "liver," "blood," and "pain." Free morphemes are the basic elements of VOSER.

b. **Bound morphemes** are fragments of words (affixes) that represent single concepts, such as "cardio-," "hepato-," "-itis," and "-megaly." Bound morphemes are used to construct complex lexemes.

c. **Complex lexemes** are single words that express more than one concept, such as "cardiomegaly," "appendicitis," and "hypertension." A complex lexeme is usually decomposed into combinations of free and bound morphemes.

d. **Phrasal Terms** are multiword terms that express one or more concepts. If a phrasal term expresses only one concept, but its "subconcepts" can be separated without loosing the combined meaning, i.e., "right lung," "upper lobe," it is called a **regular phrasal term.** If a phrasal term expresses only one concept but cannot be decomposed without loosing the combined meaning it is called a **nonregular phrasal term,** i.e., "large intestine (colon)," "side effects." Finally, if a phrasal term expresses more than one concept and can be decomposed without loosing the combined meaning it is called a **complex phrasal term,** i.e., "infiltrate in the right upper lobe," "mass adjacent to the left main bronchus."

The morphological classification is used to distinguish those vocabulary objects that need to be decomposed (molecular concepts), namely complex lexemes, regular phrasal terms, and complex phrasal terms, from those that represent indivisible single concepts (atomic concepts) (Fig. 8). This classification is important because the ED model prefers that atomic concepts be taken as attribute values in any given template. Compound objects are usually decomposed before being stored. This requirement has a significant implication on how much detail each template is able to distinguish and represent. For descriptive domains, such as physical exam and clinical history, the templates are usually rich in attributes that capture fine descriptive details, forcing the supporting vocabulary to be mainly composed of atomic concepts. On the other hand, for less descriptive domains, where conclusions and interpretations about clinical events are more important, such as problem lists and discharge summaries, the templates become much simpler, with just a few attributes being instantiated by more compound concepts.

While VOSER can clearly support both granular and coarse representations, the templates maintained by VOSER have a tendency to be as descriptive (granular) as possible. This practice, reflected by the implementation strategy that will be presented below (see Section IV), enables VOSER to support not only intervocabulary translations, but also canonical representation of concepts. In other words, VOSER needs to be at least as "granular" as the most "granular" of its client vocabularies.

Finally, another important characteristic of the MOI is its ability to represent multiple natural languages, such as English, Portuguese, and Spanish, and multiple coding schemes using a single attribute. This is possible because, in reality, any surface form is just an alphanumeric label that is used either to name a conceptual object or to describe it. Codes from any coding scheme are also just alphanumeric labels, that instead of being used by humans, are used by a computer system. Therefore, the meaning of a conceptual object is not restricted to a particular language or a particular code, and it is not dependent on a particular grammar; it is an abstract object (*42*). For instance, the concept "shortness of breath" can be expressed in English by the word "dyspnea,"

Master Object Index (partial display)

Facility	*Object ID*	*Concept ID*	*Class*	*Name*
(Master)	512	202	(template)	medication prescribed
(Master)	513	203	(attribute)	drug
(Master)	514	204	(attribute)	dose
(Master)	515	205	(attribute)	unit
(Master)	516	206	(attribute)	route
(Master)	517	207	(domain)	drugs
(Master)	518	208	(domain)	real numbers
(Master)	519	209	(domain)	units
(Master)	520	210	(domain)	routes

⇓

Template Definition (partial display)

Template	*Attribute*	*Optional*	*Multiple*
(medication prescribed)	(drug)	no	no
(medication prescribed)	(dose)	no	no
(medication prescribed)	(unit)	no	no
(medication prescribed)	(route)	no	no

⇓

Attribute Definition (partial display)

Attribute	*Type*	*Domain*
(drug)	(coded)	(drugs)
(dose)	(real)	(real numbers)
(unit)	(coded)	(units)
(route)	(coded)	(routes)

FIG. 12. Example of the Template and the Attribute Definition structures, with their relationships to the Master Object Index. Items inside parentheses are decoded for clarity.

or in Portuguese by the word "dispnéia," or in SNOMED by the code "F-20040," or in PTXT by the code "20.1.158.1.101.0.0.0" (Fig. 10).

II.2. Event Definition Structures

The Event Definition structures are the components of VOSER that store the templates created to represent the clinical data. They include the Template Definition structure and the Attribute Definition structure.

The Template Definition structure establishes the link between a template or a cluster object and its attributes (Fig. 12). The term "cluster" is used here to describe a logical group of attributes that can be reused in several different templates, because they capture a piece of information that may reappear in other domains. For example, the group of attributes used to describe the locations of the human body can be part of templates that capture different kinds of clinical observations, including radiology reports, physical exam notes, and pathology reports. While templates are by definition context specific and not shared across domains, sharing clusters helps to decrease the amount of work necessary to develop the templates and to maintain the supporting vocabulary.

Clusters identify common subdomains, helping to enforce the canonical representation of concepts across multiple domains.

The Attribute Definition structure characterizes each attribute in terms of its data type and the domain it represents (Fig. 12). The data type defines the characteristics of the attribute's value, such as numeric and alphanumeric data types and binary data types, among others. The attribute's domain corresponds to the collection of concepts that can be taken as its values. For instance, an attribute that is used to represent organs of the human body, has in its domain concepts like "kidney," "uterus," "lung," and "eye." Domains usually correspond to semantic classes, and they can be organized into different kinds of configurations that reflect their internal relationships. These configurations are represented in another set of structures, known as Configuration structures (see subsection II.4).

An important aspect of the relationship between domains and attributes is the set of restrictions imposed by the context in which the attribute is being used. When an attribute is created, its domain includes all the relevant concepts, without restriction. Whenever this attribute is used in the context of a template, its domain becomes restricted to a subdomain that is relevant to that particular context. For example, the domain of the attribute that represents organs of the human body initially contains all the concepts representing organs of the human body. When this attribute becomes a part of a template that describes chest X-ray findings, its domain becomes restricted to organs present in the thoracic cavity. An important requirement here is that any restricted domain must be totally subsumed by its parent unrestricted domain. If this is not the case (i.e., where the restricted domain ends up having additional concepts that its parent does not have), a new attribute is created to capture this newly defined domain. Domain restrictions, like the one just described, are represented using the Mapping structures (see subsection II.3).

The overall process of creating templates and attributes has fundamental implications on the VOSER content. As we mentioned before, the granularity of the vocabulary is determined by how descriptive the templates are. Therefore, if the templates become more and more specialized to capture finer details, new attributes need to be added to reflect this specialization. In addition, those attributes inherited from the parent template may have their domains more and more restricted. As a result, the overall complexity of VOSER increases, requiring special structures that are able to represent this "contextual knowledge" and at the same time are flexible enough to capture such a dynamic process (see subsection II.3).

II.3. Mapping Structures

The Mapping structures establish the connection between vocabulary objects and the "grammar" that defines how to combine these objects. The Mapping structures enable the decomposition of molecular concepts, the representation of canonical storage forms, and the domain restrictions imposed either by the

Master Object Index (partial display)

Facility	*Object ID*	*Concept ID*	*Name*
(Master)	64423	33321	right elbow fracture

⇓

Template Definition (partial display)

Template	*Attribute*	*Optional*	*Multiple*
(clinical finding)	(finding)	no	no
(clinical finding)	(body location)	yes	yes
(body location)	(body part)	no	no
(body location)	(side)	yes	no

⇓

Attribute Definition (partial display)

Attribute	*Type*	*Domain*
(finding)	(coded)	(findings)
(body part)	(coded)	(body parts)
(side)	(coded)	(sides)

⇓

Decomposition header (partial display)

Facility	*Object ID*	*Decomposition type*	*Instance ID*
(Master)	64423	(lexical decomposition)	92871

⇓

Decomposition (partial display)

Instance ID	*Template*	*Attribute*	*Value*
92871	(clinical finding)	(finding)	**(fracture)**
92871	(clinical finding)	(body part)	**(elbow)**
92871	(clinical finding)	(side)	**(right)**

FIG. 13. Example of the Decomposition structure, showing its relationships to the Template Definition, the Attribute Definition, and the Master Object Index structures. Items inside parentheses are decoded for clarity.

specialization of templates or by the instantiation of correlated domains in the same template. The first two processes use a structure called the Decomposition structure, and the last two use a structure called the Domain-Restriction structure. Despite the fact that these two structures are utilized for quite distinct functions, the underlying process of creating complete or partial instantiations of templates is a strategy applied by both of them.

The Decomposition structure enables VOSER to integrate concepts that have different granularities. This mapping is achieved by fully instantiating the appropriate template, or templates, with the molecular concept, i.e., take each atomic concept component and link it to an appropriate attribute (Fig. 13). For example, the concept "right lobe" can be decomposed into the atomic concepts "right" and "lobe." These two concepts can be taken as values of the attributes "side" and "body structure." In this example, the decomposition is used to link a molecular concept to its atomic concept components ("right lobe" = "right" + "lobe"), utilizing the surface forms as guides to the identification

Master Object Index (partial display)

Facility	*Object ID*	*Concept ID*	*Name*
(Master)	98121	55681	Parasitic thyroiditis

⇓

Decomposition header (partial display)

Concept ID	*Decomposition type*	*Instance ID*
55681	(canonical form)	272121

⇓

Decomposition (partial display)

Instance ID	*Template*	*Attribute*	*Value*
272121	(disease)	(pathologic process)	(infection)
272121	(disease)	(pathologic process type)	(parasitic)
272121	(disease)	(etiologic agent)	(Trypanosoma cruzi)
272121	(disease)	(body part)	(thyroid)

FIG. 14. Example of the Decomposition structure being used to produce a canonical form. Items inside parentheses are decoded for clarity.

of the underlying concepts. This straight decomposition is very useful when a client vocabulary that contains mainly molecular concepts is incorporated into VOSER, directing the automatic mapping process.

Composition, the reverse process, uses the same approach. Instead of breaking concepts apart, the composition process combines atomic concepts to build a molecular concept. Templates are again used as a semantic structure that guides the composition of the concepts.

The other form of decomposition that VOSER uses is called canonical decomposition. Canonical decomposition also requires the instantiation of templates from a molecular concept. The canonical decomposition process creates a unique and unambiguous representation form for the underlying molecular concept, called a canonical form. The canonical form may have little resemblance to the original surface form, and it may include new concepts that were added to assure that the meaning is unambiguous (Fig. 14). The canonical form is the computable definition of a concept, and it has a one-to-one correspondence to an MCI. In other words, a canonical form carries the meaning of the vocabulary conceptual object identified by a given MCI.

Canonical decomposition is probably the most ambitious feature of VOSER, and it certainly requires a significant amount of human expert review. However, without canonical forms, our determination to implement a MCRL is not feasible (*6*). We believe that in order to achieve complete integration, systems must exchange data using canonical forms. Ambiguity will always be present if a concept is defined outside the contexts where it is used. If VOSER fails to recognize this assumption, even after mapping the data dictionaries of the client systems, the meaning of the exchanged records will remain not explicit and sometimes incompatible.

The second Mapping structure, called the Domain-Restriction structure, creates only partial instantiations of templates. They are called partial instantiations because instead of using real atomic concepts to instantiate the attributes, valid designations of subdomains can also be used. If only designations of subdomains are used as values (see subsection II.2), the process basically narrows the attribute's domain. This process is necessary during the creation of new template specializations, i.e., restrict the domain of the attributes to reflect a context specialization. Domain restrictions are usually observed when the attributes are part of a cluster, like the example using the attribute "organs of the human body" presented in subsection II.2.

The other type of domain restriction reflects not a global contextual change, but a local one. In this particular case, atomic concepts and designations of subdomains are both used to instantiate attributes of a template, without reflecting any real structural specialization. For example, a template used to represent "prescribed drugs" will contain attributes for the "drug prescribed," and the "route used," among others. If the attribute "drug" is instantiated by a particular drug, for example "penicillin V," the domain of the attribute "route" becomes restricted to those routes that are valid for "penicillin V." The counterexample is also possible, where the route concept is fixed and it restricts the drug domain. This kind of domain restriction is particularly valuable when the template is used to guide a data entry process, as in a structured user interface. In VOSER this process supports the implementation of an intelligent user interface, and it provides valuable semantic information to the automatic mapping routines.

II.4. Configuration Structures

Configuration structures, the fourth functional component of VOSER, are used to establish both hierarchical and nonhierarchical relationships between the concepts represented in the MOI. These relationships are expressed in a structure called the Relationships structure. In addition to these logical views of the concepts, the Configuration structures are also responsible for the precise definition of the domains of the attributes. These domains are referenced by the Attribute Definition structure (see subsection II.2). The structure used to specify the domains is called the Domain Definition structure.

The Relationships structure, as its name specifies, is used to represent several kinds of semantic relationships between objects (Fig. 15). Valid configurations include simple lists of concepts (flat lists), true hierarchies (taxonomies and meronomies), tangled hierarchies, semantic networks, etc. Configurations are not limited to vocabulary objects, and they can be used to show interdependencies and specializations among templates, clusters, and attributes. Since the resulting configurations are also considered objects, it is possible to have relationships between configurations, i.e., smaller configurations can become the building blocks of larger ones.

As a general rule, VOSER initially contains the configurations necessary

Master Object Index (partial display)

Facility	*Object ID*	*Concept ID*	*Class*	*Name*
(Master)	8554	3212	Configuration	Antonyms
(Master)	12151	6511	Configuration	Template Hierarchy
(LDS Hospital)	1232	55423	Configuration	PTXT hierarchy

⇓

Relationships (partial display)

Configuration	*Object A*	*Relationship*	*Object B*
3212	(acute)	(antonym_of)	(chronic)
3212	(hypothermia)	(antonym_of)	(hyperthermia)
6511	(radiologic procedure)	(is_a)	(procedure)
6511	(X-ray)	(is_a)	(radiologic procedure)
6511	(chest X-ray)	(is_a)	(X-ray)
55423	(Gram Negative Rod) (16.1.40.1.2.0.0.0)	(grows_in)	(Urine Culture) (16.1.40.0.0.0.0.0)
55423	(Citrobacter species) (16.1.40.2.2.7.0.0)	(is_a)	(Gram Negative Rod) (16.1.40.1.2.0.0.0)
55423	(Citrobacter freundii) (16.1.40.3.2.7.1.0)	(is_a)	(Citrobacter species) (16.1.40.2.2.7.0.0)
55423	(Citrobacter diversus) (16.1.40.3.2.7.2.0)	(is_a)	(Citrobacter species) (16.1.40.2.2.7.0.0)

FIG. 15. Example of different kinds of configurations being represented in the Relationships structure. Notice that configurations are treated as objects by the Master Object Index. Items inside parentheses are decoded for clarity.

to organize its metadata structures, namely the template taxonomy and the taxonomies representing the domains of the attributes. These two configurations are used to support VOSER's functionality, and they are considered "*static knowledge objects.*" These static knowledge objects contain the categorical information that must be shared by the master and the client VOSERs.

Other configurations are usually loaded from external sources, frequently from the client vocabularies themselves. These configurations are considered "*dynamic knowledge objects,*" and they provide situational knowledge that is used to support client specific functionality. Dynamic knowledge objects are not necessarily shared with other client VOSERs; they are considered add-ons to the VOSER knowledge base. For instance, knowing that PTXT's structure is hierarchical, it is useful to recreate the PTXT hierarchy inside any client VOSER that interfaces with PTXT. Another client vocabulary, perhaps the UMLS Metathesaurus, will have its semantic network represented inside a client VOSER, and so on. In any case, client VOSERs may have different dynamic knowledge objects while the master VOSER may have none. However, if a dynamic knowledge object derived from a client vocabulary is transferred to the master VOSER, the MCIs are used to replace the original client terms. This process makes the knowledge expressed by these "external" configurations accessible to all other client VOSERs. For example, inferences that are possible due to PTXT's hierarchical organization, will become available to all

other client VOSERs that contain the concepts used to map PTXT into the master VOSER.

It is important to recognize that dynamic knowledge objects are a valuable source of knowledge, and that this knowledge can be expanded and used to enhance VOSER's functions (*40, 41*). One example of how this knowledge can be expanded is to identify the properties of the relationships, determining if they are transitive, reflexive, symmetric, etc. (*36, 41*). Once these properties are identified, they can be used to create new relationships. In terms of enhancements to the VOSER functionality, hierarchical relationships become useful when a new client vocabulary is being mapped; i.e., knowledge about narrower and/or broader concepts can help the mapping process to find the closest matches. Another example may be the adaptation of a semantic network to enhance a VOSER Browser (see subsection III.1), adding to it hypertext functionality (*45*).

The second component of the Configuration structures, called the Domain Definition structure, is a specialization of the Relationships structure. This structure defines the domains of each attribute referenced by any template known to VOSER. The domains must be true hierarachies, taxonomies ("is_a" relationship) or meronomies ("part_of" relationship) (*36*). True hierarchies ensure that the resulting configurations are not tangled (a concept having multiple parents) and not cyclic (a concept related to itself). As we will explain in Section IV, the collaborating groups are responsible for building these attribute domains. To keep the task as simple as possible, and to have the ability to share and consolidate what is produced by different groups, the domains are built from smaller subdomains that capture important classes of concepts. For example, the domain that represents medications is built by combining subdomains like antibiotics, analgesics, antipyretics, anti-inflammatories, analgesics and antipyretics, etc. This approach simplifies the overall process while providing for domain restrictions (see subsection II.3).

II.5. Maintenance Structures

The fifth functional component of VOSER is the set of Maintenance structures. These structures are responsible for keeping track of all updates to the content and to the structures maintained by VOSER. The Maintenance structures are responsible for three important functions: managing the status information of all objects defined in VOSER, maintaining a history log of the objects from their creation to their eventual deactivation, and documenting all problems that were identified and corrected. These functions are supported respectively by the Object Status structure, by the Object History structure, and by the Problem Description structure. All three structures have their entries date and time tagged. The content of these three structures is completely maintained by VOSER itself and cannot be directly altered by any user. It contains vital information that ensures the internal consistency of the overall implementation of VOSER, and it helps any auditing process that becomes necessary.

Master Object Index (partial display)

Facility	*Object ID*	*Concept ID*	*Name*
(Master)	67541	55681	South American trypanosomiasis
(Master)	98121	55681	Parasitic thyroiditis
...			
(Master)	67541	73521	South American trypanosomiasis
(Master)	98121	73524	Parasitic thyroiditis

⇓

Object Status (partial display)

Facility	*Object ID*	*Concept ID*	*Date*	*Status*	*Version*
(Master)	67541	55681	12-JAN-94	(in use)	3
(Master)	98121	55681	25-APR-94	(in use)	3
...					
(Master)	67541	55681	8-JUN-94	(inactive)	4
(Master)	98121	55681	8-JUN-94	(inactive)	4
...					
(Master)	67541	73521	19-JUN-94	(to review)	1
(Master)	98121	73524	19-JUN-94	(to review)	1

⇓

Object History (partial display)

Facility	*Object ID*	*Old Concept ID*	*New Concept ID*	*Version*
(Master)	67541	55681	73521	4
(Master)	98121	55681	73524	4

FIG. 16. Example of the Maintenance structures being used to reassign surface forms to other concepts. The version is used to link the data represented in the different structures. Items inside parentheses are decoded for clarity.

The Object Status structure stores the status history of all objects. It ultimately defines if an object is in active use or not (can be referenced by other objects), if it needs to be sanctioned or reviewed, or if it needs to be updated or maintained. In general, from the moment a new object is loaded into VOSER, it has a status assigned to it. From this initial status it undergoes a series of review phases until it is sanctioned and considered ready for use. If during its use a problem is detected, this object may end up being deactivated or eventually submitted to another review process. In addition, every time the status of a given object changes, a new sequential version number is assigned to it along with the new status (Fig. 16).

The idea of a "status" has important implications on the overall VOSER implementation and it is context dependent. A given object may have different statuses on different contexts; context being defined here as the template or the VOSER structure where the object is being used. In addition, in order to prevent conflicting status assignment, VOSER structures, namely the MOI, have status priority over the end-user templates. This priority ensures that a given object cannot be considered released for use unless it has such status assigned in the context of the MOI.

The integration of these different evolutionary phases of an object into an operational VOSER, requires checking the object status before an object can take part in any operation. Another requirement is that no object can ever be deleted if it has ever been active; an object can only be deactivated for use. This is a very important requirement because if any other object depends on the existence of a deleted object, internal inconsistencies will be created. A classical example of this problem is when a vocabulary concept that has been used by a given client vocabulary is purged from the server, invalidating all records that have been created in that client system. Another important guarantee that this requirement provides is the prevention of code reassignment and concept redefinition.

The second Maintenance structure, the Object History structure, is used to correct problems encountered after the object is released for use (Fig. 16). The Object History structure was designed to correct problems that may produce internal inconsistencies, and not simple problems that are resolved with the inactivation of the object, or one of its components. The review process tries to prevent these "corrections," but with an operational system that is continuously augmented with new medical domains, it is often the case that previous assumptions may prove to be incorrect. To ensure the interdependency between the Object History structure and the Object Status structure, the version number is used to correlate the records created.

Errors associated with conceptual objects are the ones that frequently require the use of the Object History structure. Despite the fact that any object may become incorrect, including attributes, templates, and configurations, among others, by far the most common errors are related to the vocabulary concepts. These errors appear when the intended meaning of a concept becomes inappropriate, i.e., it needs to be narrowed, or broadened, or even subdivided. In all three instances, the original concept is inactivated with the inactivation of all its surface forms. Whenever appropriate, the Object History structure is used to reassign these inactive surface forms to other concepts.

For example, in Fig. 16 the meaning expressed by the concept ID 55681 needs to be subdivided because "South American trypanosomiasis" is a generic description for the parasitic infection caused by the *Trypanosoma cruzi,* while "parasitic thyroiditis" describes the same infection but it also specifies the thyroid gland as one of the affected organs. Initially, both surface forms are inactivated using the Object Status structure, making the concept ID 55681 inactive. Next, new concept IDs are created to accommodate the new meanings (73521 and 73524). Finally, the inactive surface forms are reassigned to the new concept IDs using the Object History structure. Despite the apparent complexity of the process, we believe that it is very important to keep detailed records of such operations. These detailed records enable clients to decode previously stored data and to revalidate some of their vocabulary mappings.

The last Maintenance structure, called the Problem Description structure, is used to encode a problem that justifies an action reflected either by the Object Status structure or the Object History structure (Fig. 17). Whenever a problem

Problem Description (partial display)

Template	*Facility*	*Object ID*	*Concept ID*	*Version*	*Description*
Object Status	(Master)	67541	55681	4	(concept inactivation)
Object Status	(Master)	98121	55681	4	(concept inactivation)

FIG. 17. Example of the Problem Description structure being used to document operations presented in Fig. 16. Items inside parentheses are decoded for clarity.

is found and corrected, the user or the process that performed the correction is responsible for entering the justification. For some actions, like a change of status during the review process, the justifications are optional. In other cases, such as the inactivation of an object, or those problems that generate entries in the Object History structure, justifications are a requirement. The Problem Description structure complements the other two Maintenance structures as a valuable resource for auditing processes that trace maintenance actions. The version number and the object identifier are used to link an entry of the Problem Description structure to entries of the other two Maintenance structures.

In a client/server architecture like the one proposed for VOSER, maintenance becomes a crucial activity. As we mentioned before, an important operational premise is to guarantee the autonomy of the clients, even if it increases the maintenance costs and complexity. Therefore, the proposed Maintenance structure are a very important component, available on both master and client VOSERs. It should be remembered that since every client maintains it own vocabulary, we can expect local errors very similar to the ones just described. Even more, since clients have the capability of proposing new objects to the master, they must be ready to update their concept identifiers when the sanctioned MCIs return to them.

II.6. Application Support Structures

The last logical component of VOSER is the set of Application Support structures. These structures are specific to client VOSERs that want to incorporate the VOSER logical design into their production systems. The Application Support structures usually include structures to represent site-specific configurations and to support a wide-range of client-specific functions. The general idea behind these structures is to try to minimize the overhead created by the complexity of the VOSER design and at the same time keep what is new compatible with what is already in use. Our limited experience in this area comes from the development of the ELDR (*46*). The ELDR adopts the VOSER model, but it must be kept compatible with the data model being used by the HELP Hospital Information System.

Site-specific configurations extract from VOSER only limited sets of objects, usually necessary for data entry and for building support files. These configurations are designed to support very specific processes that require fast response

times. A good example of these configurations are "pick-lists," created to support structured data entry processes (*47*). In general, pick-lists contain a relatively small number of concepts. These concepts are represented by special surface forms customized for groups of end users. For example, a differential diagnosis pick-list specially created for the rheumatology clinic that displays mnemonics familiar to the rheumatologists. These site-specific configurations may sometimes duplicate surface forms that are already in VOSER. However, all configurations are kept fully integrated to the server. In fact, configurations cannot reference a concept that is not defined in VOSER.

Customized indexes are another example of Application Support structures. They are used to retrieve concepts and objects from VOSER using keys that are not necessary to VOSER itself. These indexes can be accessed by a broad range of applications that require access to very specific subsets of the data. For example, if the original data dictionary has a hierarchical structure, like PTXT, some applications may need hierarchical inferences to perform their functions. In this case, customized indexes can provide this functionality without affecting response time.

It is obvious that structures for supporting backward compatibility cannot be defined ahead of time. They are built according to specific needs, patching the gaps between the current data model and the VOSER model. Yet another example from the ELDR VOSER is the creation of several new logical structures just to be able to represent administrative data elements that were being stored in the PTXT data dictionary.

III. Implementation, Operation, and Maintenance Tools

An important task of the VOSER project is the identification of the necessary computer applications to both implement and maintain the server. These applications must ensure not only the overall functionality of the server, but also its internal consistency. The complexity of the underlying model requires the development of intelligent tools that offer great flexibility to the clients, without compromising the overall performance, and without violating any integrity constraints. In addition, these tools need to support both client processes that require real-time responses and also batch processes that evoke complex transactions and use large input data sets. The first kind of client processes ideally requires a graphical user interface (GUI), while the second kind requires a generic command line interface with scripting options.

Considering these premises, the computer applications we have been developing are subdivided into three main categories: client applications, collaborator applications, and administrator applications. In the next subsections we will summarize these three categories.

III.1. Client Applications

Client applications are designed to enable read-only access to VOSER by any authorized client. This category of applications has no influence on either

the implementation or the maintenance of VOSER. A first group of client applications provides functions like searching one or multiple concepts, browsing domains, and navigating through hierarchies and semantic networks, with the ability to download copies of selected concepts or domains to local files. In addition, a given client also has access to the template hierarchy, where templates could be selected to limit the scope of a given query process. Scenarios where these tools may be useful include an end user seeking all accepted surfaces forms of a given concept, or the definition of a given domain, or maybe all known relationships between a group of concepts.

A second group of client applications enables users to submit scripts of predefined operations along with data files that contain the surfaces forms or codes to be processed. This second group of client applications includes functions like the encoding of terms (parsing of terms followed by an instantiation of templates), the automatic mapping of a client vocabulary to the concepts available in VOSER (lexical and semantic translation), and the generation of text from coded input using the template structure (text generation). These advanced functions demonstrate the fundamental difference between a simple repository of medical concepts and a server supported by a MCRL. Scenarios where this functionality becomes necessary include the creation of translation tables to interface two client systems, the decodification of large sets of coded data into any desired language or coding scheme, and the generation of controlled narrative data to facilitate human understanding of a particular set of template instances.

III.2. Collaborator Applications

Collaborator applications are designed to support the review process (see Section IV), guiding the enhancement of the VOSER corpus. Their objective is to minimize the need for human intervention during these processes. C 'laborator applications include a wide range of tools that make use of the VOSER content to help identify new concepts and surface forms, as well as help design new templates. Collaborator tools assume that the user has a good understanding of the overall structure of VOSER, including its internal dependencies. This knowledge is required because the user has to make decisions about creating new objects and/or updating those already represented.

The collaborator applications include all tools created to assist a given reviewer through the two modeling phases that will be described in Section IV. For the first modeling phase, applications are designed to analyze the morphology of proposed new terms, helping to identify atomic and molecular concepts; to analyze the orthography of proposed new terms, helping to identify spelling variations and misspellings; and to analyze the written form of proposed new terms, helping to detect the presence of shortenings, nuances of capitalization, word order changes, variations of punctuation, presence of special tokens, etc. The reviewer also has access to specialized tools that help the identification of eponyms and the detection of polysemic forms. In addition, with a reasonable

inventory of concepts represented in canonical form, another set of applications can support the discovery of lexical relationships among concepts, such as synonymy, hyponymy, and hypernymy (*36*).

For the second modeling phase (see Section IV), where the templates are built, tools providing access to the inventory of clusters and attributes are mandatory. In addition, the utilization of a formal notation to express the template's logical structure is also important. We adopted the Abstract Syntax Notation One (ASN.1) (*48*), and we have been using a public domain ASN.1 compiler developed by the National Center for Biotechnology Information (NCBI) to verify the conformance of the proposed templates to the notation. The use of a standardized notation helps to focus the template development on the structure of the templates, and it adds to the overall strength of the adopted strategy. We are considering the adoption of ASN.1 as the standard structure for all the messages requesting services from the server.

III.3. Administrator Applications

Administrator applications are designed to support the overall maintenance of VOSER. The administrator applications are the only applications with privileges to change the content or the structure of VOSER. They enable not only content additions and updates, but also the implementation and maintenance of the VOSER structures themselves. Because of their importance, administrator applications should be available only to members of the review committee (see Section IV), especially to those knowledgeable about the overall VOSER architecture. The administrator applications are tightly coupled to the maintenance structures of VOSER (see subsection II.5). Records on those structures are created whenever a transaction is performed by an administrator application.

The first group of administrator applications are designed to maintain the contents of VOSER. These applications assist loading of reviewed and sanctioned material into VOSER. The loading process is responsible for generating all the auxiliary indexes that speed up content retrieval and are also responsible for generating the historical records that keep track of the evolution of the VOSER content. In addition, the loading process ensures that inconsistencies are not created either by insertions or updates, checking the output produced during the review phase. Since no human supervision is available during loading, records that seem to present a problem are flagged for a new revision. Although all these mechanisms to ensure consistency may seem redundant, tracing errors in a large dictionary with complex internal dependencies is a challenging task.

A second group of administrator applications help to maintain the VOSER logical design updates. Although these changes are not likely to be frequent, the host environment (software and hardware) of VOSER should be able to provide such tools. These low-level tools must be operated by someone with enough knowledge about VOSER's architecture and of course the necessary knowledge about the hardware and software platforms that are being used.

IV. Implementation of VOSER

We recognize that the effort to review vocabularies and to build templates is time consuming, but it is a fundamental step in the development of the content and the structure of VOSER. It is obvious that a single group cannot undertake such a task alone, both because no single group has the expertise that encompasses all possible health-related domains and because of a lack of practical needs for the end product. In addition, the development of a common vocabulary interface between any two systems demands that the parties involved have control over the data dictionaries of the systems they want to interface plus a thorough understanding of the vocabulary model and the information model being adopted by these same systems.

In reality, the VOSER implementation strategy has to be driven by real practical needs, dividing and conquering medical domains as they gain priority. The work should be performed by collaborating groups sharing a common framework, where each group works in its own domain of expertise. A common framework, in our case provided by the ED Model, helps to ensure that the end product of each group is kept open enough to be shared by other groups and also to allow future enhancements. In addition, there must be a "review committee" that helps to ensure content and structural consistency. The work of this review committee is very similar to a standards committee, where modeled domains are sanctioned to be included in the master copy of the vocabulary server.

The implementation strategy described next is in part the result of a pilot implementation conducted from October of 1993 to April of 1994. Only the review process phases are presented, since they are the most important aspects of the VOSER implementation.

During a first review phase, the participating groups are invited to formalize the representation of their domains of expertise using the framework supported by the ED model. In a second review phase, selected members from the participating groups, organized as a committee, unify the material produced by their individual groups. An important requirement of this implementation strategy is that each group must be engaged in some form of system development, or in a research project that will benefit from the existence of VOSER. This requirement ensures that the material produced by each group reflects a real practical need and tests the proposed framework against a broad range of applications.

The groups participating in the project have to initially get acquainted with the ED model and its framework. This "training" step starts with a sequence of meetings where the ED model is described in some detail, and all groups attend them together. After these joint meetings, the groups start to work alone trying to represent their domains of interest. At this stage, they work under the supervision of experienced participants, or members of the review committee.

For each domain chosen by one of the participating groups, two modeling phases occur. The first modeling phase focuses on the lexicographic analysis of the selected domains (*43*), and it ultimately results in new "subdomain vocabularies" being added to VOSER. During this first phase, each group

selects potential sources of medical terms relevant to their domains. These sources may include *controlled medical vocabularies,* such as the UMLS Metathesaurus (*9*), SNOMED International (*44*), ICD-9 (*38*), and PTXT (*30*); *narrative data,* such as discharge summaries, problem lists, description of clinical history and physical exams, and radiology reports; and *records from coded medical databases.*

If the potential source is a controlled medical vocabulary, subsets of this vocabulary can be isolated and loaded into VOSER. Whenever a controlled vocabulary is loaded, we try to preserve the lexical information available, the existing mappings to other vocabularies, as well as any native hierarchical and nonhierarchical configurations.

If the potential source is a collection of narrative documents ("free text"), the processes used include the isolation of the words with their respective frequencies, the generation of keyword-in-context lists (kwic) (*41, 43*), the identification of frequent sequences of words, and the consultation of on-line dictionaries. These processes combined try to simulate a "thesaurus discovery" approach (*49*), but they clearly rely upon human expert review. A large collection of narrative documents from a given domain provides an excellent source of surface forms used to express the concepts relevant to that domain (*6, 14, 41*), and, in addition, it enables the identification of nonstandard abbreviations (truncations), domain-specific acronyms, common misspellings, and expressive or frequently used noun phrases. All relevant concepts and surface forms isolated from the narrative documents are also loaded into the VOSER structures.

In our site, given the availability of the HELP System, a common scenario for this first phase is the existence of a controlled vocabulary that offers a detailed coverage of the domain being modeled (PTXT Data Dictionary), plus a reasonably large collection of structured (coded) and/or unstructured (free-text) records pertaining to the same domain. When this happens, both the controlled vocabulary and the narrative documents are analyzed using the processes just described.

The second modeling phase expands the output of the first phase, emphasizing the development of the templates (data structures) that "describe" the domain expressed by the new vocabulary. These templates become the "grammar" that enables the decomposition of molecular concepts using the Mapping structures (see subsection II.3) and also the metadata structures that formalize the creation of database records in this domain.

Two important factors influencing this second phase are: (a) the overall organization of the selected domain, i.e., its structural complexity, and (b) the intended use of the information being captured by the templates. For well-structured domains, where relevant information is not transmitted using narrative descriptions, the required attributes of the templates can be identified by either analyzing current database records or by looking at the paper forms used to collect these data. Examples of such domains include results of laboratory exams, prescription of drugs, and charting of vital signs. Structured domains usually have well-defined controlled vocabularies that do not require extensive

human review. In addition, some well-structured domains are the focus of standardization committees (*50*) whose work can be adapted to become storage templates.

For domains that are mainly available in an unstructured form, the effort to create the templates is more complex. In this case, the review of the vocabulary used has a special value. Analyzing which medical concepts are used and how they are linked to each other end up revealing clues of which attributes are necessary to represent that domain. In order to define these "implicit" attributes, the concepts are assigned to one or more pertinent semantic classes. The assignment of semantic classes is a rather subjective process. In order to decrease the subjectivity, each class has a formal definition that tries to clarify its meaning and scope. In addition, whenever possible, we try to reuse semantic classes already defined in the literature (*9, 20*) or to reuse the semantic classes that have been defined for other domains previously modeled.

The second factor that influences the development of the templates, i.e., the intended use of the information being represented, is what ultimately defines the complexity of the templates and the granularity of the associated vocabulary. In general, if the client system is designed to manipulate information and generate new data, like an expert system, the granularity of its vocabulary is more likely to be coarse, with a predominance of molecular concepts. However, if the client system's main function is to collect and report information, like a hospital information system, its vocabulary has a tendency of being more atomic, designed to capture primitive data elements. In any case, the server needs to reflect the different granularities, and to allow for the composition and decomposition of the molecular concepts. In general, attributes are defined by the atomic concepts, while templates reflect the logical organizations expressed by the molecular concepts.

Discussion

The VOSER project is clearly an ambitious endeavor and its success will depend not only on how well it is designed, but also on how well it is implemented and maintained. An important factor that has contributed to its design, and that will certainly spawn future implementations, is the usefulness that a comprehensive repository of medical concepts has for the development and integration of clinical information systems (*51, 52*) and the implementation of the computer-based medical record (*5, 53*). In other words, the need for a VOSER is clearly present.

Considering the need for a VOSER, we propose three fundamental rules for its design and implementation. First, it is necessary to recognize the importance of having a collaborative process. Collaboration is the best way to accumulate the expertise necessary to formalize the terminology of all the different medical domains. With collaboration, we also obtain valuable practical knowledge both from the intricacies of the domains and the applications that the collaborating groups aim to support. It is very important to reuse and to learn what is

already available, always trying to provide backwards compatibility. If real collaboration exists, where many different groups voice their needs and propose solutions, the whole design process is very likely to be kept open. An open design is necessary to merge complementary models, and to resolve previous misconceptions. Another important characteristic of an open model is its potential extensibility.

The second important rule is to identify the applications or processes requiring a VOSER before the logical design takes place. A clear identification of the services to be supported by the VOSER is necessary to define its scope and its content. In other words, it is not realistic to try to support a myriad of services reflected by numerous potential clients and their respective theoretical needs. We believe that a well-conceived medical language representation model is capable of sustaining a wide variety of applications. However, we also believe that the representation of extralinguistic knowledge and the complexity of some specialized methods are not just "additional" tasks. As we stated at the beginning, the context of this project is limited to a VOSER that enables the exchange of patient data and medical decision logic, and special emphasis is given to the mapping of different vocabularies into a common logical structure. Therefore, some of the premises emphasized here may not apply to a VOSER designed to support automatic indexing of documents, or to a VOSER created to assist natural language understanding applications, among others.

Another important effect of this second rule is that utilizing VOSER to support real applications helps to define its operational requirements. Successful implementations using the same VOSER infrastructure are good indicators of its functionality, and can eventually attract the interest of other developers.

The third and final rule is the agreement upon a common representation model. A common representation model provides a "standard" interface to VOSER, forcing different systems to "speak" a common "language" using the same vocabulary. A common model also provides a single framework that different groups use to model their domains; i.e., the groups are themselves invited to speak a common language. As we mentioned before, the model adopted by VOSER may be considered just an interface formalism by some clients, but it may also be partially or totally incorporated into applications developed at other client sites. The need for a common representation model is clearly stated in the literature (*5, 6*), and other models could have been used to implement a similar vocabulary server (*10, 32–35*).

In addition to these three basic requirements, we have gained valuable experiences from the ongoing develoment of the ELDR and ACIS and the pilot implementation of VOSER. We added new structures to the ED model following important suggestions given by these groups, and we had a chance to strengthen the proposed design by adjusting it to different environments. However, problems were also identified during these interactions with other groups. One of the most prevalent problems was the difficulty in sharing the proposed conceptual model (ED model). Another important problem was the interaction between groups. While each group could clearly see what elements were necessary to

support their particular needs, they were sometimes reluctant to introduce new structures in order to support another group's needs. As the process evolves and we gain more experience with it, we hope to be able to solve these problems. We also foresee new challenges arising from the existence of operational applications based on VOSER's design, namely ACIS and ELDR.

In terms of the future of VOSER, we would like to see the creation of a regional "VOSER committee" that would take over and direct the implementation of an operational server. Not forgetting the practical need that drives the VOSER initiative, members of the committee would be responsible for introducing the VOSER concepts into their own environments. With appropriate funding and organizational support, this committee would be responsible for coordinating new groups to extend the VOSER content, developing new applications, and granting access to potential clients. In addition, the committee would also be responsible for sharing the experiences with other groups, eventually contributing to the implementation of national and international vocabulary servers.

ACKNOWLEDGMENTS

The authors thank all the groups that participated in this project, namely Medical Informatics (University of Utah Health Sciences Center, LDS Hospital, and Primary Children's Medical Center), Nursing Informatics, Intermountain Health Care, 3M Health Care Systems, Applied Medical Informatics, and Veterans Administration's Information Systems Center of Salt Lake City. This project was partially supported by Grant 1 R03 HS 08053-01 from the Agency for Health Care Policy and Research. Roberto A. Rocha is supported by a scholarship from the National Council for Scientific and Technological Development (CNPq), Secretary for Science and Technology, Brazil.

REFERENCES

1. HUFF, S. M., CRAIG, R. B., GOULD, B. L., CASTAGNO, D. L., AND SMILAN, R. E. Medical data dictionary for decision support applications. *In* "Proceedings of the Eleventh Symposium on Computer Applications in Medical Care," pp. 310–317. IEEE, Los Angeles, 1987.
2. LINNARSSON, R., AND WIGERTZ, O. The data dictionary—A controlled vocabulary for integrating clinical databases and medical knowledge bases. *Methods Informat. Med.* **28**(2), 78–85 (1989).
3. JOHNSON, S. B., CIMINO, J. J., FRIEDMAN, C., HRIPCSAK, G., AND CLAYTON, P. D. Using metadata to integrate medical knowledge in a clinical information system. *In* "Proceedings of the Fourteenth Symposium on Computer Applications in Medical Care" (R. A. Miller, Ed.), pp. 340–344. IEEE Computer Society Press, Los Angeles, 1990.
4. BERSON, A. "Client/Server Architecture." McGraw–Hill, New York, NY, 1992.
5. Board of Directors of the American Medical Informatics Association. Standards for medical identifiers, codes, and messages needed to create an efficient computer-stored medical record. *J. Am. Med. Informat. Assoc.* **1**(1), 1–7 (1994).
6. EVANS, D. A., CIMINO, J. J., HERSH, W. R., HUFF, S. M., AND BELL, D. S. (for the Canon Group). Toward a medical-concept representation language. *J. Am. Med. Informat. Assoc.* **1**(3), 207–17 (1994).
7. ROCHA, R. A., HUFF, S. M., AND HAUG, P. J. Implementation of a controlled medical vocabulary server. *In* "1994 Spring Congress Final Program and Abstract Book" (T. C. Rindfleisch, Ed.), pp. 77. American Medical Informatics Association, 1994.
8. ROCHA, R. A., HUFF, S. M., HAUG, P. J., KOEHLER, S., AND WANG, P. Towards a representa-

tion scheme for clinical data. *In* "1993 Spring Congress Final Program" (M. G. Kahn, Ed.), pp. 84. American Medical Informatics Association, 1993.
9. LINDBERG, D. A. B., HUMPHREYS, B. L., AND MCCRAY, A. T. The Unified Medical Language System. *Methods Informat. Med.* **32**(4), 281–291 (1993).
10. GABRIELI, E. R. A new electronic medical nomenclature. *J. Med. Syst.* **13**(6), 355–373 (1989).
11. PRATT, A. W. Medicine, computers, and linguistics. *Adv. Biomed. Eng.* **3**, 97–140 (1973).
12. BLOIS, M. S. "Information and Medicine—The Nature of Medical Descriptions." Univ. of California Press, Berkeley, CA, 1984.
13. WINGERT, F. Medical linguistics: Automated indexing into SNOMED. *CRC Crit. Rev. Med. Informat.* **1**(4), 335–403 (1987).
14. SAGER, N., FRIEDMAN, C., AND LYMAN, M. S. "Medical Language Processing—Computer Management of Narrative Data." Addison–Wesley, Reading, MA, 1987.
15. MUSEN, M. A. Dimensions of knowledge sharing and reuse. *Comput. Biomed. Res.* **25**(5), 435–467 (1992).
16. ROSSI MORI, A., THORNTON, A. M., AND GANGEMI, A. An entity-relationship model for a European machine-dictionary of medicine. *In* "Proceedings of the Fourteenth Symposium on Computer Applications in Medical Care" (R. A. Miller, Ed.), pp. 185–189. IEEE Computer Society Press, Los Angeles, 1990.
17. MASARIE, F. E., MILLER, R. A., BOUHADDOU, O., GIUSE, N. B., AND WARNER, H. R. An interlingua for electronic interchange of medical information: Using frames to map between clinical vocabularies. *Comput. Biomed. Res.* **24**, 379–400 (1991).
18. RECTOR, A. L., NOWLAN, W. A., AND KAY, S. Foundations for an electronic medical record. *Methods Informat. Med.* **30**(3), 179–86 (1991).
19. CÔTÉ, R. A., AND ROTHWELL, D. J. The classification–nomenclature issues in medicine: A return to natural language. *Med. Informat. (London)* **14**(1), 25–41 (1989).
20. EVANS, D. A. Pragmatically-structured, lexical–semantic knowledge bases for unified medical language systems. *In* "Proceedings of the Twelfth Symposium on Computer Applications in Medical Care" (R. A. Greenes, Ed.), pp. 169–173. IEEE Computer Society Press, Los Angeles, 1988.
21. FRIEDMAN, C., HRIPCSAK, G., JOHNSON, S. B., CIMINO, J. J., AND CLAYTON, P. D. A generalized relational schema for an integrated clinical patient database. *In* "Proceedings of the Fourteenth Symposium on Computer Applications in Medical Care" (R. A. Miller, Ed.), pp. 335–339. IEEE Computer Society Press, Los Angeles, 1990.
22. FILLMORE, C. J. The case for case. *In* "Universals in Linguistic Theory," (E. Bach and R. Harms, Eds.), pp. 1–90. Holt, Rinehart, and Winston, New York, NY, 1968.
23. SCHANK, R. C., AND COLBY, K. M. (Eds.) "Computer Models of Thought and Language." Freeman, 1973.
24. CANFIELD, K., BRAY, B., HUFF, S., AND WARNER, H. Database capture of natural language echocardiology reports. *In* "Proceedings of the Thirteenth Symposium on Computer Applications in Medical Care" (L. C. Kingsland III, Ed.), pp. 559–563. IEEE Computer Society Press, Los Angeles, 1989.
25. CANFIELD, K., BRAY, B., AND HUFF, S. Representation and database design for clinical information. *In* "Proceedings of the Fourteenth Symposium on Computer Applications in Medical Care" (R. A. Miller, Ed.), pp. 350–353. IEEE Computer Society Press, Los Angeles, 1990.
26. FU, L. S., BOUHADDOU, O., HUFF, S. M., SORENSON, D. K., AND WARNER, H. R. Toward a public domain UMLS patient database. *In* "Proceedings of the Fourteenth Symposium on Computer Applications in Medical Care" (R. A. Miller, Ed.), pp. 170–174. IEEE Computer Society Press, Los Angeles, 1990.
27. ROCHA, R. A., ROCHA, B. H. S. C., AND HUFF, S. M. Automated translation between medical vocabularies using a frame-based interlingua. *In* "Proceedings of the Seventeenth Symposium on Computer Applications in Medical Care" (C. Safran, Ed.), pp. 690–694. McGraw–Hill, New York, NY, 1993.

28. PRYOR, T. A., GARDNER, R. M., CLAYTON, P. D., AND WARNER, H. R. The HELP System. *J. Med. Syst.* 7(2), 87–101 (1983).
29. PRYOR, T. A., WARNER, H. R., GARDNER, R. M., CLAYTON, P. D., AND HUANG, P. J. The HELP System development tools. *In* "Implementing Health Care Information (H. F. Orthner and B. I. Blum, Eds.), pp. 365–383. Springer-Verlag, New York, NY, 1989.
30. KUPERMAN, G. J., GARDNER, R. M., AND PRYOR, T. A. "HELP: A Dynamic Hospital Information System." Springer-Verlag, New York, NY, 1991.
31. WARNER, H. R., HAUG, P. J., LINCOLN, M., WARNER JR., H. R., SORENSON, D., AND FAN, C. Iliad as an expert consultant to teach differential diagnosis. *In* "Proceedings of the Twelfth Symposium on Computer Applications in Medical Care" (R. A. Greenes, Ed.), pp. 371–376. IEEE Computer Society Press, Los Angeles, 1988.
32. ROSSI MORI, A., BERNAUER, J., PAKARINEN, V., RECTOR, A. L., ROBBÈ, P., CEUSTERS, W., HURLEN, P., OGONOWSKI, A., AND OLESEN, H. Model for representation of terminologies and coding systems in medicine. *In* "Proceedings of the Seminar: Opportunities for European and U.S. Cooperation in Standardization in Health Care Informatics," Geneva, September 1992.
33. RECTOR, A. L., NOWLAN, W. A., AND KAY, S. Unifying medical informatics using an architecture based on descriptions. *In* "Proceedings of the Fourteenth Symposium on Computer Applications in Medical Care" (R. A. Miller, Ed.), pp. 190–194. IEEE Computer Society Press, Los Angeles, 1990.
34. CIMINO, J. J., HRIPCSAK, G., JOHNSON, S. B., AND CLAYTON, P. D. Designing an introspective, multipurpose, controlled medical vocabulary. *In* "Proceedings of the Thirteenth Symposium on Computer Applications in Medical Care" (L. C. Kingsland III, Ed.), pp. 513–518. IEEE Computer Society Press, Los Angeles, 1989.
35. EVANS, D. A., ROTHWELL, D. J., MONARCH, I. A., LEFFERTS, R. G., AND CÔTÉ, R. A. Toward Representations for Medical Concepts. *Med. Decision Making* **11**(4), S102–S108 (Suppl.) (1991).
36. CRUSE, D. A. "Lexical Semantics." Cambridge Univ. Press, Cambridge, UK, 1987.
37. RECTOR, A. L., NOWLAN, W. A., AND GLOWINSKI, A. Goals for concept representation in the GALEN project. *In* "Proceedings of the Seventeenth Symposium on Computer Applications in Medical Care" (C. Safran, Ed.), pp. 414–418. McGraw–Hill, New York, NY, 1993.
38. United States National Center for Health Statistics. "International Classification of Diseases, Ninth Revision, with Clinical Modifications." The Center, Washington, DC, 1980.
39. American Medical Association. "Physician's Current Procedural Terminology." American Medical Association, Chicago,IL, 1991.
40. CIMINO, J. J., CLAYTON, P. D., HRIPCSAK, G., AND JOHNSON, S. B. Knowledge-based approaches to the maintenance of a large controlled medical terminology. *J. Am. Med. Informat. Assoc.* **1**(1), 35–50 (1994).
41. EVENS, M. W. (Ed). "Relational Models of the Lexicon—Representing Knowledge in Semantic Networks." Cambridge Univ. Press, Cambridge, UK, 1988.
42. ROSSI MORI, A., GALEAZZI, E., GANGEMI, A., PISANELLI, D. M., AND THORNTON, A. M. The transfer of medical concepts in a multilingual environment. *In* "Proceedings of the Workshop: Issues in Transfer of Medical Data, Information and Knowledge," Brussels, pp. 1–14, 1990.
43. BENNETT, P. A., JOHNSON, R. L., MCNAUGHT, J., PUGH, J. M., SAGER, J. C., AND SOMERS, H. L. "Multilingual Aspects of Information Technology." Gower, Brookfield, VT, 1986.
44. CÔTÉ, R. A., ROTHWELL, D. J., PALOTAY, J. L., BECKETT, R. S., AND BROCHU, L. (Eds.) "The Systematized Nomenclature of Medicine: SNOMED International." College of American Pathologists, Nothfield, IL, 1993.
45. CIMINO, J. J., ELKIN, P. L., AND BARNETT, G. O. As we may think: The concept space and medical hypertext. *Comput. Biomed. Res.* **25**(3), 238–263 (1992).
46. 3M Health Information Systems. "Technical Design of Clinical Data Repository Structure and Services for Enterprise Lifetime Data Repository (ELDR)." 3M Health Information Systems, Salt Lake City, UT, 1994.
47. HUFF, S. M., PRYOR, T. A., AND TEBBS, R. D. Pick from thousands: A collaborative processing

model for coded data entry. *In* "Proceedings of the Sixteenth Annual Symposium on Computer Applications in Medical Care" (M. A. Frisse, Ed.), pp. 104–108, McGraw–Hill, New York, NY, 1992.

48. International Organization for Standardization. "International Standard ISO/IEC 8824: Specification of Abstract Syntax Notation One (ASN.1), Second Edition" International Organization for Standardization, Geneva, Switzerland, 1990.
49. FRAKES, W. B., AND BAEZA-YATES, R. (Eds.) "Information Retrieval—Data Structures & Algorithms." Prentice–Hall, Englewood Cliffs, NJ, 1992.
50. MCDONALD, C. J. Standards for the Electronic Transfer of Clinical Data: Progress, Promises, and the Conductor's Wand. *In* "Proceedings of the Fourteenth Symposium on Computer Applications in Medical Care" (R. A. Miller, Ed.), pp. 9–14, IEEE Computer Society Press, Los Angeles, 1990.
51. PRYOR, T. A., AND HRIPCSAK, G. Sharing NLM's: An Experiment between Columbia–Presbyterian and LDS Hospital. *In* "Proceedings of the Seventeenth Symposium on Computer Applications in Medical Care" (C. Safran, Ed.), pp. 399–403. McGraw–Hill, New York, NY, 1993.
52. WONG, E. T., PRYOR, T. A., HUFF, S. M., HAUG, P. J., AND WARNER, H. R. Interfacting a stand-alone expert system with a hospital information system. *Comput. Biomed. Res.* **27**(2), 116–129, 1994.
53. DICK, R. S., AND STEEN, E. B. (Eds.) "The Computer-Based Patient Record—An Essential Technology for Health Care." National Academy Press, Washington, DC, 1991.