Terminology Tools: State of the Art and Practical Lessons

J. J. Cimino
Department of Medical Informatics, Columbia University, New York, NY, USA

Summary
Objectives: As controlled medical terminologies evolve from simple code-name-hierarchy arrangements, into rich, knowledge-based ontologies of medical concepts, increased demands are placed on both the developers and users of the terminologies. In response, researchers have begun developing tools to address their needs. The aims of this article are to review previous work done to develop these tools and then to describe work done at Columbia University and New York Presbyterian Hospital (NYPH).
Methods: Researchers working with the Systematized Nomenclature of Medicine (SNOMED), the Unified Medical Language System (UMLS), and NYPH's Medical Entities Dictionary (MED) have created a wide variety of terminology browsers, editors and servers to facilitate creation, maintenance and use of these terminologies.
Results: Although much work has been done, no generally available tools have yet emerged. Consensus on requirement for tool functions, especially terminology servers is emerging. Tools at NYPH have been used successfully to support the integration of clinical applications and the merger of health care institutions.
Conclusions: Significant advancement has occurred over the past fifteen years in the development of sophisticated controlled terminologies and the tools to support them. The tool set at NYPH provides a case study to demonstrate one feasible architecture.

Keywords:
Terminology, Vocabulary, Semantics, Hospital Information Systems

Method Inform Med 2001; 40: 298–306

* This paper was originally presented at the International Medical Informatics Association Working Group 6 meeting, in 1999. It has been modified to reflect changes since that time in the field at large and at the New York Presbyterian Hospital.

1. Introduction

Concept-oriented terminologies have been proposed and developed by many different researchers over the past decade [1]. All of these works share a knowledge-based approach to term representation. Specifically, they make use of constructs such as semantic networks or conceptual graphs to provide explicit relationships among terms in order to express definitional information about the terms. For example, the term "Aspirin Tablet" might be linked to the term "Medication" via an "is-a" hierarchical relation, to the chemical term "Acetylsalicylic Acid" via a "has-ingredient" relation, and to the drug form term "Tablet" via a "has-dosage-form" relation. Taken in the aggregate, this set of relations serves as a formal definition for "Aspirin Tablet" [2].

As terminologies evolve from simple code-name-hierarchy arrangements, into rich, knowledge-based representations of medical concepts, increased demands are placed on both the developers and users of the terminologies. In response, researchers have begun developing tools to address their needs. In this paper, I will review published literature on knowledge-based terminology tools (including browsers, editors and servers) and describe the tools in use at Columbia University and New York Presbyterian Hospital.

2. Terminology Browsers

A prerequisite to more sophisticated terminology tools is the availability of terminology browsers that allow users to search and navigate through the terminology without necessarily providing editing capabilities. Most computer applications with controlled terminologies provide users with some form of term look-up; to my knowledge, the first tool specifically created for interacting with a large, complex standard terminology was Lowe's Medical Subject Headings (MeSH) browser, called MicroMeSH [3]. This program provided a "search assistant" with a rich set of lexical look-up facilities and a "tree walker" to support navigation of MeSH's complex structure, including terms with as many as seven different contexts (hierarchical locations).

With the creation of the Unified Medical Language System (UMLS), the navigation problem increased by an order of magnitude, since there were now multiple complex terminologies, all interwoven in a Metathesaurus and associated with a semantic network [4]. The result was a Hypercard stack called Meta-1 that displayed the names, codes, contexts, of concepts as they appeared in specific terminologies, as well as the relationships among the concepts; the relationships served as hyperlinks among the concept descriptions [5]. Since then, many UMLS developers and users have created browsing tools, ranging from query facilities using commercial products (such as Microsoft Access) to more sophisticated knowledge-based systems [6].

3. Terminology Editors

The terminologies of the past could have been (and often were) created and maintained with simple word processors. This approach will not suffice for the kinds of terminologies that are now emerging. Inserting a new term, or changing an existing one, requires creation of links between

Received March 19, 2001
Accepted May 21, 2001

terms in complex ways. This requires precise pointers and referential integrity. It would not do, for example, to link "Aspirin Tablet" via a "has-dosage-form" relation to the term "Table" because of a typographical error. A sophisticated editing tool would notice if "Table" was not actually a valid term or, if it was, that it was not an appropriate choice. Similarly, inserting a term into a hierarchy requires inheritance of attributes and checks for introduction of cycles.

Tuttle, Sperzel and colleagues at Lexical Technologies were faced with a variety of complex tasks during the course of creating and maintaining the Unified Medical Language System (UMLS) [7]. They approached this by defining a database schema for representing the UMLS knowledge and then characterizing associated procedures needed to produce the UMLS content [8]. This approach led to the development of the MEME (Metathesaurus Enhancement and Maintenance Environment) tools [9].

At about the same time, Mays and colleagues, at Ontyx (formerly Lexstar) began using an existing system for managing description logic, called K-Rep, to represent a prototype terminology called InterMED [10] and a larger terminology being developed at Kaiser-Permanente [11]. Their tool provided capabilities for automatically completing term definitions and classifying them using inheritance and subsumption inferences. Based on that experience, they went on to develop a system called Ontylog that was specifically designed for editing and classifying description-logic-based terminologies. (Mays E, personal communication).

Campbell and colleagues were also interested in description-logic-based terminology and applied their approach to the Convergent Medical Terminology project between Kaiser-Permanente, the Mayo Clinic and SNOMED. They had similar requirements to Mays, with the added complication of needing to coordinate editing changes among disparate groups. As a result, they created the Gálapagos suite of tools [12] and were able to demonstrate that the description logic approach could support unification of potential editing conflicts [13].*

4. Terminology Servers

Terminologies of the past could be browsed with simple word processors and queried with simple relational databases. These approaches have proven to be inadequate for more complex terminologies. With the standardization of terminologies, disparate applications are called upon to use them in similar ways. As a result, client-server architectures have emerged for providing users with ways to browse and search terminologies, and for servicing the needs of clinical application that use terminologies for data entry or data display. Servers are called on to provide lists of terms matching users' input, translations from one terminology to another, and class-based queries.

Rocha and colleagues have developed the VOSER vocabulary server to support the terminology needs of the HELP system at LDS Hospital in Salt Lake City [14]. They adopted an "event definition" model to represent patient data with coded terms. VOSER does not distinguish between terms and relations but rather treats both as concepts that can be interrelated in any number of ways. This provided a great deal of flexibility and expressivity. VOSER has become the basis for the 3M Healthcare Data Dictionary [15].

Rector, et al. have developed a Terminology Server for the GALEN project [16]. This server supports the expression of input terms into concepts represented in the GRAIL formalism, the mapping of terms into the one of several natural (i.e., European) languages, and the conversion of terms to codes in one of several standard terminologies. Several research groups have successfully used the Terminology Server for their own terminological work [17, 18] Recently, the GALEN tools have been made widely available by the *Open*-GALEN foundation [19].

Gennari and colleagues, as part of the InterMed project (not related to the aforementioned K-Rep project), developed an architecture for distributed terminology editing with what they referred to as an ontology server [10], written in the knowledge representation language Ontolingua [20]. The server supported both a Web-based browser/editor and an application program interface (API) for interacting directly with other applications needing access to the terminology.

Because it provides a variety of information about a large number of terminologies, the UMLS Metathesaurus [21] has been an ideal candidate for inclusion in terminology servers. The National Library of Medicine provides the UMLS Knowledge Source Server (KSS), that offers command line, API and Web-based interfaces to support browsing and querying [22]. The development of an application to use the KSS to map user terminology (including misspellings and incomplete terms) to terminologies used by medical information resources is under way [23].

Many other researchers have exploited the UMLS to create servers of their own. Hersh and Leone created the SAPHIRE Server, to support string-matching techniques for matching user input to UMLS concepts [24]. Nadkarni created the Concept Locator, a relational database of the Metathesaurus to support client-based Boolean queries for term look-up [25]. Burgun and colleagues created the Model for Assistance in the Orientation of a User within Coding Systems (MAOUSSC), a Web-based terminology server that supports UMLS term look-up with semantic restrictions, to support the encoding of medical procedures [26]. Dierks and colleagues describe an architecture for a coordinated suite of servers, one of which is the Vocabulary Server for UMLS term look-up [27]. Finally, Lexical Technologies (the maintainers of the UMLS) have created their own server to support term look-up, called Metaphrase [28]. Elkin and colleagues have successfully incorporated Metaphrase as a reusable component of their problem list generation system [29].

Given the broad interest in terminology servers, the Object Management Group (OMG) published a Request for Proposals for the drafting of specifications for Lexicon Query Services [30]. 3M Health

* As an historical note, Lexical Technologies and Ontyx have merged to form Apelon, which also makes Campbell's tools available.

Systems responded with a 175-page proposal that included detailed specifications for the data model and interface to be incorporated in a server that met the complete requirements of the OMG. Thankfully, Chute et al. have boiled this down to a simplified set of server requirements [31]. They then consider the notion of a "clinical terminology server" that serves the specific purpose of mapping clinician data input to terms in the clinical terminology. From the general requirements, they have gleaned a set of nine "desiderata" that would be necessary for a clinical terminology server: word normalization, word completion, target terminology specification, spelling correction, lexical matching, term completion, semantic locality, term composition and term decomposition.

5. Experience at Columbia University and New York Presbyterian Hospital – The Medical Entities Dictionary

Much of the work reported in the informatics literature to date deals with descriptions of approaches and pilot projects for tool development. Very little has been written regarding the complex assembly of tools needed to support enterprise-wide terminology needs. The remainder of this paper attempts to address this deficiency by describing the terminology work of my colleagues and myself at Columbia University. This work has provided key technologies in the development of the clinical information system at New York Presbyterian Hospital, a system that spans in-patient and outpatient settings at several facilities, in use for the past thirteen years.

What began as a knowledge-based terminology effort [32] has grown into a repository called the Medical Entities Dictionary (MED) [33]. Conceptually, the MED is a frame-based semantic network, in which each MED concept has some number of named slots with values that may be literal values or pointers to other MED concepts. It currently contains over 67,000 concepts, with terms drawn from those used in laboratory, pharmacy, radiology, and billing systems. It includes 206,000 synonyms, 100,000 hierarchical relations, 167,000 other semantic relations, and 139,000 mappings to other terminologies, including the UMLS, ICD9-CM, and LOINC. The relationships in the network provide definitional knowledge about the individual terms; Fig. 1 depicts some examples of this knowledge.

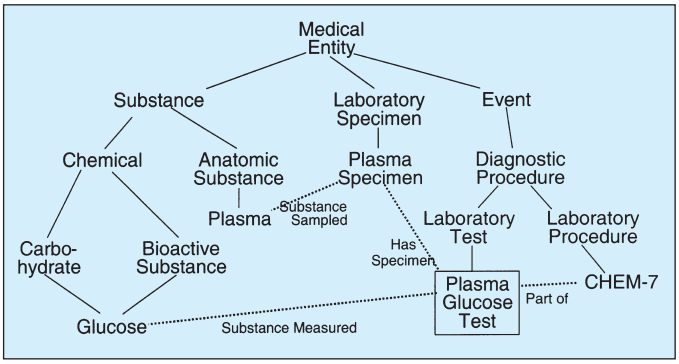

Fig. 1 A sample from the Medical Entities Dictionary of concepts and their interrelations. The term in the box (Plasma Glucose Test) is shown in relation to its parents in the is-a hierarchy (solid lines) and via non-hierarchical semantic links (broken lines) to other terms in the network.

The MED was constructed to serve the primary purpose of a repository for codes and terms used by clinical applications to represent data in the clinical data repository [34]. The knowledge included in the MED was originally intended to support intelligent vocabulary management tools. However, as the repository grew and the data in it were reused in a variety of ways, the MED knowledge was reused as well. In many cases, the MED served as a convenient repository for additional knowledge used by various applications, and so it grew to serve as a tight link between clinical applications and the terminologies used by them [35].

The first version of the MED was created and maintained using a commercial object-oriented knowledge engineering environment, called KEE (Intellicorp, Mountain View, CA) [36]. The KEE

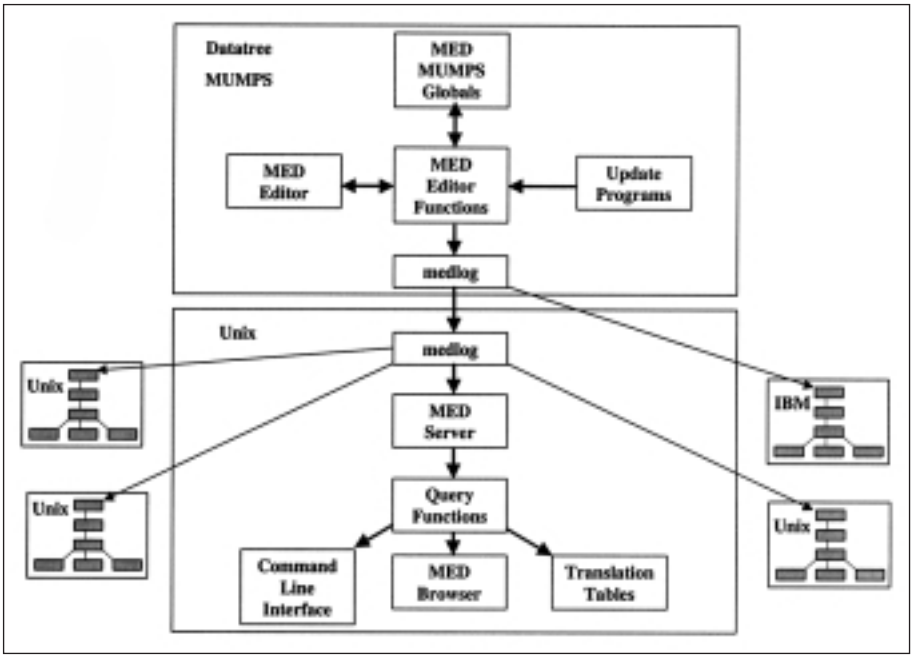

Fig. 2 MED Architecture. The "gold standard" MED is maintained on a personal computer in MUMPS, using a set of diting functions. Changes are recorded in a "medlog" that is transferred to an IBM mainframe and to a single Unix system for dissemination to other systems. Each system supports a server that, in turn, supports browsing and querying.

Fig. 3 The MUMPS MED Editor. The squares on the left are editing functions (corresponding to the function keys on an IBM PC-AT keyboard; they can be activated with function key strokes or mouse clicks), the top panel shows the MED hierarchy for the concept of interest, and the bottom panel shows its the frame-based representation.

environment provided rich capabilities for creating sophisticated browsing and editing tools, including automated classification of terms in the MED [37]. However, by 1991 the MED had grown to over 4,000 terms and neither the PC-based nor the mainframe-based versions of KEE, nor any other commercial product, had sufficient capacity to contain the MED's knowledge base. We were therefore compelled to develop our own editors, browsers, and servers.

The original system was developed in MUMPS on a personal computer. Changes in MED content are reflected in change files called *medlogs* that are then distributed as updates to other systems (including several Unix midrange computers and an IBM mainframe) where they are applied as updates to the content of MED servers. The servers, in turn, support a variety of functions including queries, browsers, and translation tables. Fig. 2 shows an overview of this architecture. Recently, a Unix version has been developed as an eventual replacement. Currently, either of these systems (but not both at once) can serve as the "gold standard".

5.1 The MUMPS Environment

The MUMPS version of the MED exists on a personal computer, running MUMPS (Datatree, Waltham, MA) under a Windows-NT (Microsoft, Redmond, WA) environment. The MED is stored in two MUMPS globals – one that is indexed by *medcode* (the unique identifiers for MED concepts) and contains all the slot information for each concept, and a second one that is indexed by slot code and provides information about the slots, such as slot name, value type, etc.

The globals are changed by a set of editing functions that carry out basic operations, such as creating new entities, renaming entities, creating slots, and adding and removing slot values (including hierarchical relationships). The editing functions carry out integrity checks on all proposed changes and provide a rich set of error messages when illegal changes are proposed. Illegal changes include: creating or renaming a MED concept with a name that already exists, adding a parent-child link that would create a cycle, removing the sole parent link of an entity (orphaning), removing a slot value that does not exist, adding a slot value that already exists, removing an inherited slot value, and adding a link to an inappropriate concept (for example, attempting to link a test to a specimen through the "substance-measured" relationship). The editing functions record successful changes in the current medlog file.

Editing functions can be called in two ways: using the MED Editor or as part of an update program. The MED Editor, shown in Fig. 3, is a character-based browser that simulates a graphical windowing environment. Browsing can be carried out with mouse clicks and lexical look-ups, while editing is carried out with function key actions. The update programs operate in either a batch mode or semiautomatic mode to modify some part of the MED to keep it synchronized with external terminologies from other systems (e.g., laboratory and pharmacy) [38].

Since all of these programs use the same set of editing functions, one medlog is maintained for all changes made to the MED. When a significant number of changes have been made to the MED, or some update needs to be disseminated to support other applications, the medlog is given a sequence number (currently medlog 415) and transferred to one of the Unix systems and to the IBM mainframe.

5.2 The Unix Environment

Once the medlog has been transferred to the Unix environment, it is disseminated to several other Unix systems, each of which contains the identical MED environment and content. On each system, the medlog is applied to a static version of the MED, producing a new static version that serves as input to the Unix MED server.

The Unix server consists of a data structure and a set of C library functions. The data structure resides in shared memory and remains loaded and ready for use by the library functions. These functions provide access to all of the MED content and support a variety of look-up techniques, including exact match, automatic stemming, substring match, synonym matching, and keyword synonyms. Automatic stemming allows partial entry of terms, such as "cong hear f" to find terms with names such as "Congestive Heart Failure". Substring matching returns terms that have a character string appearing anywhere in their name (e.g., "icillin"). Synonym matching retrieves terms that have each of the search

words or stems in their name or in any of their synonyms. Keyword matching returns terms that match all of the search words or any synonyms of the search words. For example, because the MED includes the knowledge that the words "kidney" and "renal", and the words "disease" and "disorder", are often interchangeable [39], searching on "renal disease" is equivalent to the Boolean expression "(renal or kidney) and (disease or disorder)".

The Unix MED library functions are typically accessed by other programs, such as a natural language processing system called MedLEE [40] and the MED browser AccessMED [41]. Shown in Fig. 4, this browser runs in an X-window environment and provides users with access to all of the information about MED concepts in a point-and-click graphical browser. The semantic network of the MED is displayed as a directed acyclic graph of hierarchical links; the user has the option of displaying any or all semantic links. The AccessMED browser has been incorporated into a clinical application to allow physicians and nurse practitioners to enter problems, medications, and allergies into patient records [42]. The architecture has also supported the development of Web-based MED browsers, an example of which is shown in Fig. 5.

The library functions are also available through a command-line program called qrymed. This program allows users to include look-up functions in non-C environments, such as Perl scripts. For example, the command:

qrymed -find renal

returns the medcodes for all terms with the word "renal" in their names (currently,

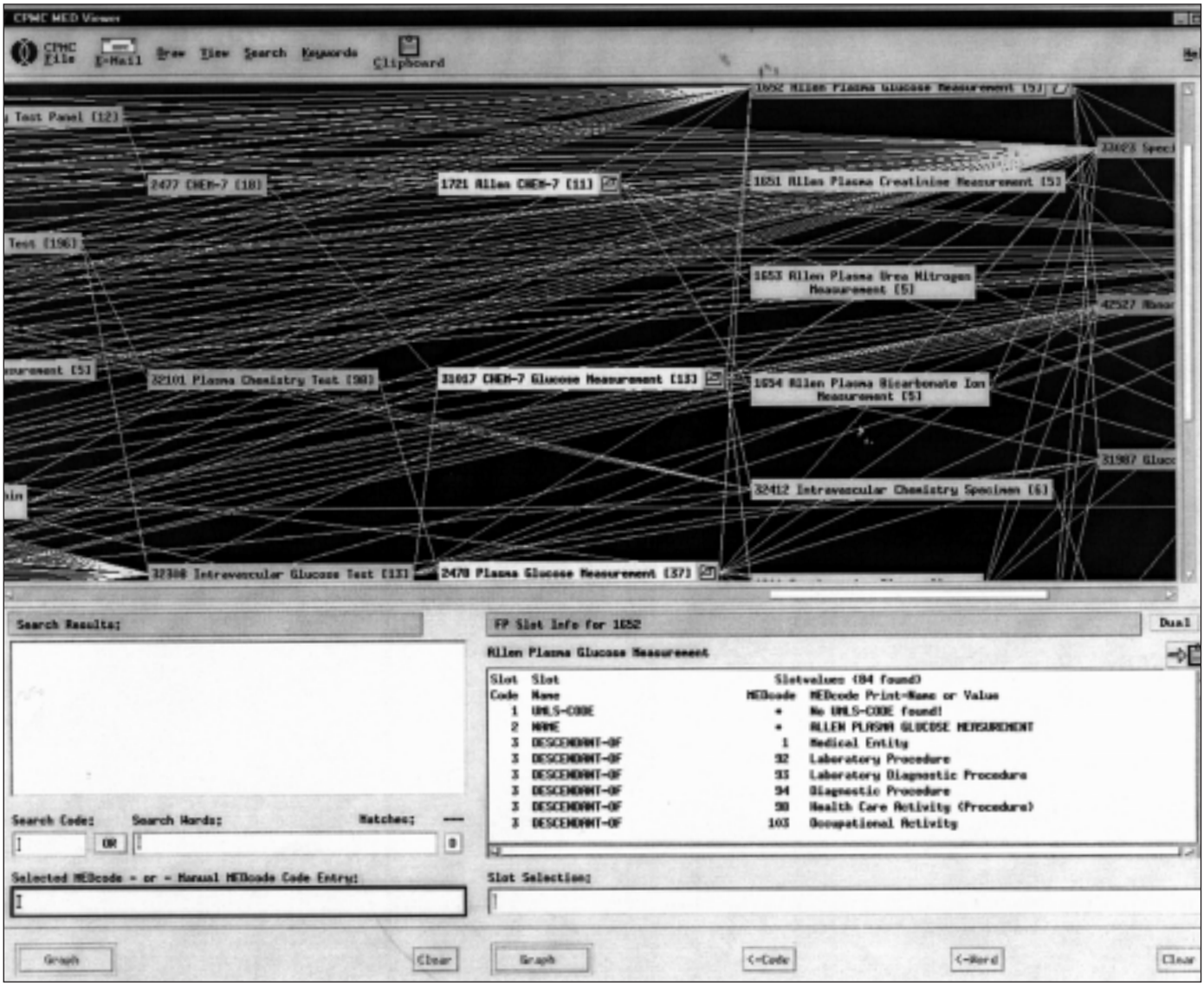

Fig. 4 AccessMED. The top panel displays the semantic network, the lower left panel provides search capabilities, and the lower right panel displays the frame-based description of the current concept of interest.

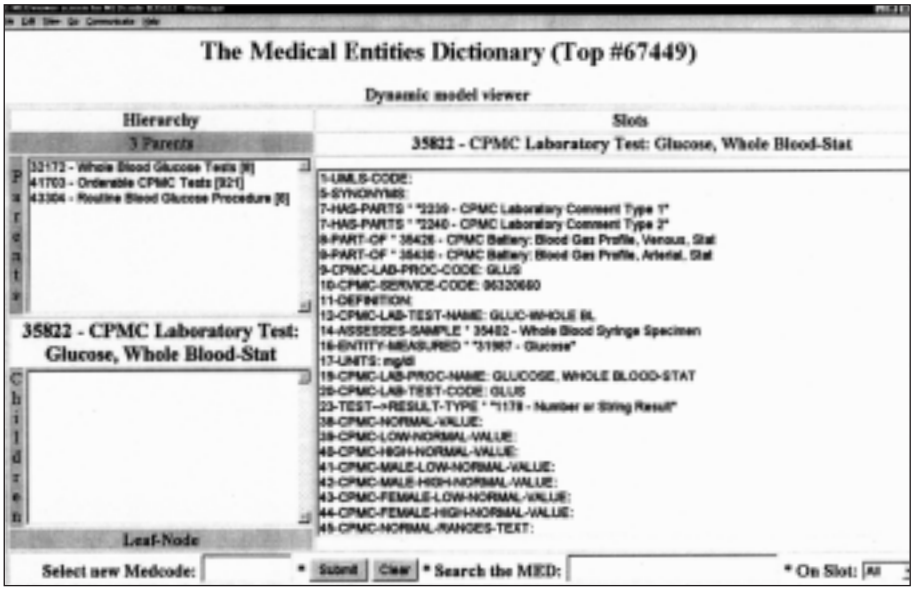

Fig. 5 A Web-based MED browser. The left side displays the parents and children (none, in this case) of the concept of interest, the right side displays the frame-based description of the current concept of interest, and the bottom provides search capabilities.

348 terms). The commands can also be chained, using the unix "pipe". For example, the command:

qrymed -find renal | qrymed -desc | qrymed -nm

will return the names of all descendants of any term containing the word renal in its name (currently 586 terms). Table 1 shows a partial list of the MED library functions.

The MED is also used in the Unix environment to support a number of clinical applications, including uploading data to the clinical repository [43] and Web-based reviewing of these data [44]. In some cases, the MED libraries are used directly; in other cases, *translation tables* are used. A translation table is typically a two-column table consisting of a list of terms with one column for the medcodes and one column for the values of a particular slot. The tables are generated nightly, using a simple class-based (that is, descendant) query to the MED. The tables are used during data upload by looking up the name or code for each datum as supplied by the ancillary system, and replacing it in the upload message with the medcode. The tables are used during data display by looking for the display names associated with the medcodes used to code the data. These tables can be incorporated directly into programs to provide more efficient access to MED information, and can be transferred for use on systems that do not have MED servers, such as personal computers used to upload data from ancillary systems.

Table 1
Options for qrymed look-up functions

-allslts:	lists all the slots, with their names.
-scd:	returns the slotcode with string x as its name.
-snm:	returns the name of slotcode x.
-srecip:	returns the reciprocal of slotcode x.
-stype:	returns the type for slotcode x.
-cd:	returns the medcode with string x as its name (exact match).
-find:	lists medcodes that have string x in their names (pattern match).
-nm:	returns the name for medcode x.
-pnm:	returns the print name for medcode x.
-par:	lists the parents of medcode x.

5.3 The Mainframe Environment

The MED mainframe architecture is similar to the Unix architecture in that it contains a server, query facilities and translation tables. On the mainframe, however, the server is the DB2 database management system, with queries carried out as native SQL statements in programs that require MED access. Fig. 6 shows how the semantic and hierarchical information in the MED is represented using relational tables.

Translation table creation is carried out at the time that the medlog changes are applied to the DB2 database. This is accomplished by indicating to the update program which tables are related to which slots. When these slots are changed in any way by the update process, the corresponding translation tables are generated automatically.

5.4 The Web-based Unix MED Editor

As noted above, an editor has been created in the Unix environment. This editor uses the MED server and a user interface similar that shown in Fig. 5. As changes are made to the MED, a change file is generated that contains updated versions of affected MED concepts. The editor uses these updates in combination with information returned by the server to produce the view of the MED shown to the user. This approach works well until the change file becomes large, which causes the perfor-

mance of the system to degrade. At that point, the user can commit the changes and produce a new medlog file, which is then used in the same manner as the one produced by the MUMPS system.

6. Discussion

As the sophistication of medical terminologies has evolved, so too have the capabilities of the tools needed for the maintenance and use of the terminologies. A number of commercially available systems are now being offered to support concept-based terminologies. Our thirteen years of experience at Columbia University and the New York Presbyterian Hospital has provided us with some understanding of the capabilities necessary for successful deployment of controlled terminologies for use in production clinical systems. Among these are the requirements for editors, browsers, and servers.

Editing a terminology is challenging when that terminology includes a rich knowledge base for concept representation. The task is further complicated when it involves the coordination of terms generated by one diverse set of applications and used by another diverse set of applications. We have had some assistance with these tasks through the use of intelligent editing tools that can, for example, assist with automated classification of terms. We have also been able to use batch processes to take updates from a variety of sources and apply them in an efficient and timely manner. However, as the complexity of our task grows with hospital mergers and expansion of clinical systems, the size and complexity of the MED has grown as well. As a result, the single-user editor will no longer be adequate. We are therefore moving to modify our Web-based editor to support distributed editing tasks.

Our terminology server capabilities were developed in direct response to local requirements, before any industry standards were available. According to Chute et al. [31], the CORBAMed terminology server specification [30] identifies eight mandatory and three optional services. The MED servers in the Unix environment support all of these capabilities.

Chute et al. go on to identify desiderata for clinical terminology servers that would be used for data entry by clinicians. For the most part, our server provides these capabilities, with some notable differences. In particular, we do not support advanced lexical techniques such as word normalization. We have found that rich keyword synonymy serves somewhat the same purpose. We also do not support spelling correction; however, word stemming compensates somewhat for spelling errors.

The "Target Terminology Specification" desideratum is not supported directly; however, we can carry out this function by making use of a more powerful function: class restriction. Class restriction is used to limit the domain over which lexical searches are performed. For example, when attempting to enter a patient problem, if the user types "penicillin", the retrieval can be limited to descendants of the class "Patient Problem", thus retrieving terms like "Penicillin Allergy" while avoiding all of the medication terms that would be retrieved if the restriction was not in force. Class restriction can be used to retrieve terms from target terminologies (to satisfy the desideratum) by limiting the retrieval to descendants of the class that corresponds to the terminology, such as "ICD9-CM Diseases".

For us, terminology services have required not only a library of query facilities but also mechanisms for coordination of editing and distribution functions. A single MED server is not practical for our diverse environment and thus our distributed approach is necessary. Furthermore, we have found that the ability to generate class-based translation tables, in a way that is coordinated with the update process, has greatly simplified the work of application developers who would otherwise have difficulty making their systems work with the MED.

The MED is not intended as a general purpose or standard terminology; the MED content is relevant only for our institution. However, the tools and architecture are not institution-specific (for example, we have successfully used them to browse the UMLS and SNOMED-RT), and the lessons learned are relevant for understanding the challenges involved with maintaining and using knowledge-based terminologies in a wide variety of production clinical information systems. Additional work is needed to create distributed editing capabilities, and to find ways to "MED-enable" commercial clinical information systems being used at our institution.

Entities

1
2
3
4

Slots

10	Name
20	UMLS
30	Part-of
40	Specimen

Entity-Slots

1	10
2	10
2	20
2	30

Entity/Slot/Values

1	10	Entity
2	10	C0001
2	40	1234
2	50	mg/dl

Ancestry

1	1
1	2
1	3
2	3

Fig. 6 Relational Tables used to represent the MED in DB2. The Entities and Slots tables lists all valid MED Codes and Slot Codes, respectively. The Entity-Slot table indicates which terms have which slots, while the Entity/Slot/Values table provides the value for those slots, if any. The Ancestry table contains the transitive closure of the parent-child relationships.

Acknowledgments
The MED, its editors, browsers, and servers have been the work of many people, including: Barry Allen, Randy Barrows, Gai Elhanan, Bruce Forman, Nilesh Jain, David Wajngurt and Adam Wilcox. In addition, I am indebted to George Hripcsak, Steve Johnson, Soumitra Sengupta, Bob Sideli, and Paul Clayton for their contributions to the philosophy and design of the MED.

References

1. Cimino JJ. From data to knowledge through concept-oriented terminologies: experience with the Medical Entities Dictionary. JAMIA. 2000; 7 (3): 288-97.
2. Campbell KE, Das AK, Musen MA. A logical foundation for representation of clinical data. JAMIA 1994; 1: 218-32.
3. Lowe HJ., Barnett GO. MicroMeSH: A microcomputer system for searching and exploring the National Library of Medicine's Medical Subject Headings (MeSH) vocabulary. In: "Proceedings, 11th Annual Symposium on Computer Applications in Medical Care" (W.W. Stead, Ed.), pp. 717-20. IEEE Computer Society, Washington DC 1987.
4. Tuttle MS, Sherertz D, Erlbaum M, Olson N and Nelson SJ. Implementing Meta-1: the first version of the UMLS Metathesaurus. In: Kingsland LC, (ed).: Proceedings of the Thirteenth Annual Symposium on Computer Applications in Medical Care. IEEE Computer Society Press, New York 1989: 483-7.
5. Sperzel D, Erlbaum M, Fuller L, et al. Editing the UMLS Metathesaurus: Review and Enhancement of a Computed Knowledge Source. In: Miller RA, ed. Proceedings of the Fourteenth Annual Symposium on Computer Applications in Medical Care, November 4-7, Washington, DC; IEEE Computer Society Press, New York 1990: 631-40.
6. Schulz S, Romacker M, Udo H. Knowledge engineering in the UMLS. Studies in Health Technology and Informatics 2000; 77: 701-5.
7. Tuttle MS, Sherertz DD, Erlbaum MS, Sperzel WD, Fuller LF, Olson NE, Nelson SJ, Cimino JJ, Chute CG. Adding Your Terms and Relationships to the UMLS Metathesaurus. In: Clayton PD, ed.: Proceedings of the Fifteenth Annual Symposium on Computer Applications in Medical Care; Washington DC November, 1991: 219-23.
8. Sperzel WD, Tuttle MS, Olson NE, Erlbaum MS, Saurez-Munist O, Sherertz DD, Fuller LF. The Meta-1.2 engine: a refined strategy for linking biomedical vocabularies. In: Frisse ME, ed.: Proceedings of the Sixteenth Annual Symposium on Computer Applications in Medical Care; Baltimore, MD; November, McGraw-Hill, New York 1992: 304-8.
9. Suarez-Munist ON, Tuttle MS, Olson NE, Erlbaum MS, Sherertz DD, Lipow SS, Cole WG, Keck KD, Davis AN. MEME-II supports the cooperative management of terminology. JAMIA 1996; 2 (suppl): 84-8.
10. Shortliffe EH, Patel VL, Cimino JJ, Barnett GO, Greenes RA. An experiment in collaboration among medical informatics research laboratories. Artificial Intelligence in Medicine 1998; 12 (2): 97-123.
11. Mays E, Weida R, Laker M, et al. Scalable and expressive medical terminologies. JAMIA 1996; 3 (suppl): 259-63.
12. Campbell KE, Cohn SP, Chute CG, Rennels G, Shortliffe EH. Galapagos: computer-based support for evolution of a convergent medical terminology. JAMIA 1996; 3 (suppl.): 269-73.
13. Campbell KE, Cohn SP, Chute CG, Shortliffe EH, Rennels, G. Scalable methodologies for distributed development of logic-based Convergent Medical Terminology. Methods of Information in Medicine 1998: 37 (4-5): 426-39.
14. Rocha RA, Huff SM, Haug PJ, Warner HR. Designing a controlled medical vocabulary server: the VOSER project. Computers and Biomedical Research 1994; 27 (6): 472-507.
15. http://www.mmm.com/market/healthcare/his/product/hems/fctsheet/hdd.htm
16. Rector AL, Solomon WD, Nowlan WA, et al. A terminology server for medical language and medical information systems. Methods of In-formation in Medicine 1995; 34 (1-2): 147-57.
17. Wagner JC, Solomon WD, Michel P-A, Juge C, Baud RH, Rector AL, Scherrer J-R. Multilingual natural language generation as a part of a medical terminology server. In: Kaihara S, Greenes RA, eds. Proceedings of the World Congress on Medical Informatics – Medinfo '95; Vancouver, Canada; Healthcare Computing and Communications Canada, Edmonton, Alberta 1995: 100-4.
18. Sherrer JR, Spahni S. Healthcare information system architecture (HISA) and its middleware modules. JAMIA 1999; 6 (suppl.): 935-9.
19. http://www.opengalen.org
20. Gennari JH, Oliver DE, Pratt W, Rice J, Musen MA. A Web-based architecture for a medical vocabulary server. In: Gardner RM, ed.: Proceedings of the Nineteenth Annual Symposium on Computer Applications in Medical Care; New Orleans, LA; October-November, Hanley & Belfus, Philadelphia 1995: 275-9.
21. Sherertz DD, Tuttle MS, Cole W, Erlbaum M, Olson N, Nelson SJ. A HyperCard implementation of Meta-1: the first version of the UMLS Metathesaurus. In: Kingsland LW, ed. Proceedings of the Thirteenth Annual Symposium on Computer Applications in Medical Care. Washington, D.C. November 1989: 1017-8.
22. McCray AT, Razi AM, Bangalore AK, Browne AC, Stavri PZ. The UMLS Knowledge Source Server: a versatile Internet-based research tool. JAMIA 1996; 3 (suppl.): 164-68.
23. McCray AT, Loane RF, Browne Allen C, Bangalore AK. Terminology issues in user access to Web-based medical information. JAMIA 1999; 6 (suppl.): 107-11.
24. Hersh W, Leone TJ. The SAPHIRE Server: a new algorithm and implementation. In: Gardner RM, ed.: Proceedings of the Nineteenth Annual Symposium on Computer Applications in Medical Care; New Orleans, LA; October-November, Hanley & Belfus, Philadelphia 1995: 858-63.
25. Nadkarni PM. Concept Locator: a client-server application for retrieval of UMLS Metathesaurus concepts through conplex Boolean query. Computers and Biomedical Research 1997; 30: 323-36.
26. Burgun A, Denier P, Bodenreider O, et al. A Web terminology server using UMLS for the description of medical procedures. JAMIA 1997; 4 (5): 356-63.
27. Dierks MM, Sands DZ, Safran C. Re-engineering the process of surgical informed consent. JAMIA 1999; 6 (suppl.): 731-5
28. Tuttle MS, Olson NE, Keck KD, et al. Metaphrase: an aid to the clinical conceptualization and formalization of patient problems in healthcare enterprises. Methods of Information in Medicine 1997; 37 (4-5): 373-83.
29. Elkin PL, Mohr DN, Tuttle MS, et al. Standardized problem list generation, utilizing the Mayo Canonical Vocabulary embedded within the Unified Medical Language System. JAMIA 1997; 4 (suppl.): 500-4.
30. http://www.omg.org/docs/corbamed/97-01-02.htm
31. Chute CG, Elkin PL, Sherertz DD, Tuttle MS. Desiderata for a clinical terminology server. JAMIA 1999; 6 (suppl.): 42-6.
32. Cimino JJ, Hripcsak G, Johnson SB and Clayton PD. Designing an introspective, controlled medical vocabulary. In: Kingsland LW, ed. Proceedings of the Thirteenth Annual Symposium on Computer Applications in Medical Care. Washington, DC; November, IEEE Computer Society Press, New York 1989: 513-8.
33. Cimino JJ, Clayton PD, Hripcsak G, Johnson SB. Knowledge-based Approaches to the Maintenance of a Large Controlled Medical Terminology. Journal of the American Medical Informatics Association 1994; 1 (1): 35-50.
34. Johnson S, Friedman C, Cimino JJ, Clark T, Hripcsak G, Clayton PD. Conceptual data model for a central patient database. In: Clayton PD, ed.: Proceedings of the Fifteenth Annual Symposium on Computer Applications in Medical Care; Washington, DC; November 1991: 381-5.
35. Cimino JJ. From data to knowledge through concept-oriented terminologies: experience with the Medical Entities Dictionary. JAMIA 2000; 7 (2): in press.
36. Cimino JJ, Hripcsak G, Johnson SB, Friedman C, Clayton PD. Prototyping a Vocabulary Management System in an Object-Oriented Environment. In: Timmers T, Blum BI, eds.: Proceedings of the International Medical Informatics Association Working Conference on Software Engineering in Medical Informatics, North-Holland Press, Amsterdam, October 1990: 429-39.
37. Cimino JJ, Hripcsak G, Johnson SB, Friedman C, Fink DJ, Clayton PD. UMLS as Knowledge Base – A Rule-Based Expert System Approach to Controlled Medical Vocabulary Management. In: Miller RA, ed.: Proceedings of the Fourteenth Annual Symposium on Computer Applications in Medical Care; Washington, D.C.; November, 1990: 175-180.
38. Cimino JJ, Johnson SB, Hripcsak G, Sideli RV, Fink DJ, Friedman C, Clayton PD. One Year's Experience with the Unified Medical Language System (UMLS) in Academia and Patient Care. In: Lun KC, Degoulet P, Piemme TE, Rienhoff, eds.: MEDINFO 92, Elsevier, North-Holland Press, Amsterdam; September 1992: 1501-5.

39. Cimino JJ. Auditing the Unified Medical Language System with semantic methods. JAMIA 1998; 4: 41-51.
40. Friedman C, Johnson SB, Forman B, Starren J. Architectural requirements for a multipurpose natural language processor in the clinical environment. In: Gardner RM, ed. Proceedings of the Nineteenth Annual Symposium on Computer Applications in Medical Care; New Orleans, LA; October-November, Hanley & Belfus, Philadelphia 1995: 347-51.
41. Allen BA, Barrows RC, Desai N. AccessMED, a tool for browsing, searching and maintaining a large controlled medical vocabulary. JAMIA. 1996; 3 (suppl.): 940.
42. Barrows RC, Allen BA, Smith K, Arni VV, Sherman E. A decision-supported outpatient practice system. JAMIA 1996; 3 (suppl.): 792-6.
43. Forman BH, Cimino JJ, Johnson SB, Sengupta S, Sideli R, Clayton P. Applying a controlled medical terminology to a distributed production clinical information system. In: Gardner RM, ed.: Proceedings of the Nineteenth Annual Symposium on Computer Applications in Medical Care; New Orleans, LA; October-November, Hanley & Belfus, Philadelphia, 1995: 421-5.
44. Hripcsak G, Cimino JJ, Sengupta S. WebCIS: large scale deployment of a Web-based clinical information system. JAMIA. 1999; 6 (suppl.): 804-8.

Correspondence to:
James J. Cimino, M.D.
Department of Medical Informatics
Vanderbilt Clinics Room 558
Columbia-Presbyterian Medical Center
622 West 168th Street
New York, New York 10032
E-mail: jjc7@columbia.edu