A Web–Based Architecture for a Medical Vocabulary Server

John H. Gennari, Diane E. Oliver, Wanda Pratt, James Rice and Mark A. Musen

Section on Medical Informatics
Knowledge Systems Laboratory
Stanford University
Stanford, CA 94305-5479, U.S.A

email: gennari, oliver, pratt, musen@camis.stanford.edu; rice@ksl.stanford.edu
URL: http://camis.stanford.edu/projects/intermed-web/vocab/

Abstract

For health care providers to share computing resources and medical application programs across different sites, those applications must share a common medical vocabulary. To construct a common vocabulary, researchers must have an architecture that supports collaborative, networked development. In this paper, we present a web-based server architecture for the collaborative development of a medical vocabulary: a system that provides network services in support of medical applications that need a common, controlled medical terminology. The server supports vocabulary browsing and editing and can respond to direct programmatic queries about vocabulary terms. We have tested the programmatic query-response capability of the vocabulary server with a medical application that determines when patients who have HIV infection may be eligible for certain clinical trials. Our emphasis in this paper is not on the content of the vocabulary, but rather on the communication protocol and the tools that enable collaborative improvement of the vocabulary by any network-connected user.

1. A Medical Vocabulary Server

Computer-based medical applications make use of controlled vocabularies to maintain consistent usage of terms. Early computer-based patient record systems, such as the Computer-Based Ambulatory Record (COSTAR) [1], The Medical Record (TMR) [2], and the HELP system [3], created their own controlled vocabularies independently to serve local clinical needs. Simultaneously, a number of standard medical terminologies became widespread, such as the International Classification of Diseases (ICD-9-CM) and Current Procedural Terminology (CPT). By the 1980's, the proliferation of controlled medical vocabularies was impeding progress in data and system integration, and in response, the National Library of Medicine embarked on the Unified Medical Language System (UMLS) project to facilitate translation among vocabularies [4].

As health-care information management becomes more automated, the demand for a sharable, controlled medical vocabulary that adequately covers clinical content and that can be integrated with other standard vocabularies will increase. To meet this demand, we need to provide a reliably maintained vocabulary as well as easy network access into the vocabulary. In effect, we should build a *vocabulary server*. The services that we envision providing to the medical community include: (1) tools for building and maintaining the vocabulary, (2) tools that allow users to browse and search through the vocabulary and (3) a protocol for a direct programmatic interface into the vocabulary for use by medical applications.

For a medical vocabulary to be widely accepted and used, we believe it should be built *collaboratively*, and its design should be driven by real application needs. Any vocabulary service developed at a single location by a single group of researchers will necessarily be biased toward the needs and the medical applications at that site. However, if an architecture is established that allows collaborative development of the vocabulary and if network services are provided that allow easy access to that vocabulary, then we believe that the system would have a much greater likelihood of serving a wider range of health care needs.

Research on the collaborative development of shared resources has been underway in our laboratory for a number of years as part of the Knowledge Sharing Technology Project [5]. As part of this effort, we have developed a general-purpose, web-based *ontology server* for editing and browsing ontologies. Formally, an *ontology* is a partial specification of a universe of discourse [6,7]; for our purposes, a medical ontology is the set of terms and relationships that comprise a model of medical concepts. In this paper, we describe our first-generation implementation of the ontology server and our use of this server as a prototype for a medical vocabulary server. By storing our medical vocabularies as ontologies and by using the existing ontology server, we can focus on issues of collaborative content development and application–server interactions, rather than low-level implementation problems.

Figure 1 shows a schematic view of our approach. The right-most box depicts the general-purpose ontology server, with the medical vocabulary shown as one ontology within the library of ontologies. The ontology server uses the world-wide web communication standard of the hypertext transfer protocol (HTTP); this choice aims to reach the widest possible audience of users. Any user familiar with common web-browsing tools such as NetScape Navigator™ or NCSA Mosaic can either browse through a vocabulary, or can build and maintain a vocabulary. In addition to

0195–4210/95/$5.00 © 1995 AMIA, Inc.

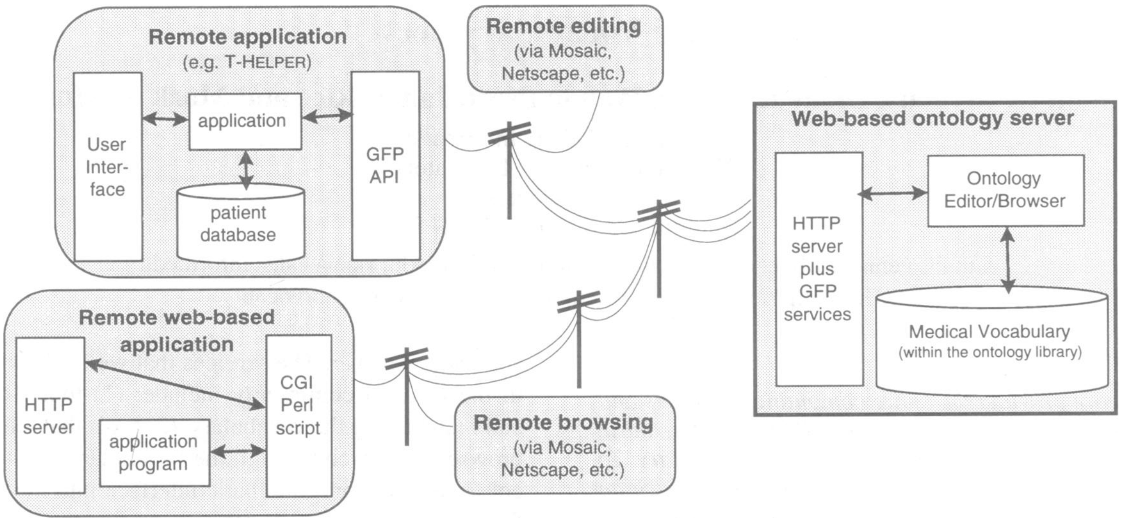


Figure 1. A schematic view of interactions with the ontology server. The server can process either direct queries for information from applications by means of the generic frame protocol (GFP), or page requests from web-browsing tools such as Mosaic and Netscape.

editing and browsing services, we also provide a direct, programmatic interface to the information in a vocabulary. The ontology server supplies this service by responding to direct queries according to the *generic frame protocol* [8], a portable interface for querying knowledge bases.

In addition to describing our longer-term goals for the design of a medical vocabulary server, we also include in this paper a concrete example of server–client interaction with a medical application, the T-HELPER system. This example application demonstrates the need for a direct programmatic interface with a server.

2. THE T-HELPER APPLICATION

The T-HELPER medical application is an outpatient computer-based record system for patients with human immunodeficiency virus (HIV) infection [9]. It includes a decision-support system that encourages enrollment of patients in clinical trials and that assists clinicians with protocol-based therapy. The system is currently installed and undergoing evaluation at two local county-operated AIDS clinics.

To enroll a patient in a clinical trial, the T-HELPER system must compare patient data with the eligibility criteria associated with that clinical trial. Typically, a clinical trial has between 15 and 50 criteria for determining eligibility. Many of these criteria have to do with laboratory-test results and the patient's history of medication. An example of an eligibility criterion is "The patient may not be currently undergoing treatment with an antiretroviral drug." To test for this type of criterion, T-HELPER must be able to classify drugs and determine whether or not a given drug is an "antiretroviral" drug. Before the vocabulary server was available, the system simply included a large classification tree of all the drugs that are (or might be) mentioned in a therapy trial. To assess eligibility for a particular patient, the system retrieved that patient's medication information from a patient database and compared this information to the descriptions in the eligibility criteria. If the patient's list of current medications included zidovudine, it would find that zidovudine is a type of antiretroviral drug and would conclude that the patient is ineligible for this clinical trial.

Thus, without a vocabulary server, the T-HELPER system must include a locally maintained vocabulary hierarchy. To be used effectively for eligibility determination, this vocabulary must cover (1) all vocabulary terms used by authors of clinical trials and (2) all drug names used in the clinic patient databases. As one might imagine, keeping this drug list up-to-date is an on-going and daunting task, especially as experimental trials and drug names change over time. The current vocabulary contains just over 1000 drug classifications.

A better approach would be for the eligibility determination program to query and retrieve information from an authenticated, up-to-date medical vocabulary server. While this does not solve any of the difficult issues of vocabulary coverage and maintenance, it would allow T-HELPER programmers to delegate these problems to a separate vocabulary service. Additionally, the vocabulary content (the drug names) would then be available to other medical applications. Therefore, the cost of building and maintaining the drug vocabulary could be amortized across all applications that use that vocabulary.

3. THE ONTOLOGY SERVER

The ontology server shown in Figure 1 has been developed as part of the Knowledge Sharing Technology Project, a general research effort to support knowledge sharing and reuse. The server, its ontology library and its ontology editor can be used by anyone with a web-browsing tool by connecting to the location (URL):

http://www-ksl-svc.stanford.edu:5915/

This server can respond to queries for vocabulary information, exactly as needed by the T-HELPER application. In addition to supporting T-HELPER's vocabulary needs, the Knowledge Sharing Technology Project also provides tools that enable us to work toward our longer-term goal: the collaborative development of a centralized medical vocabulary. In the next three subsections, we describe features of the ontology server that support distributed, collaborative development of consensus ontologies.

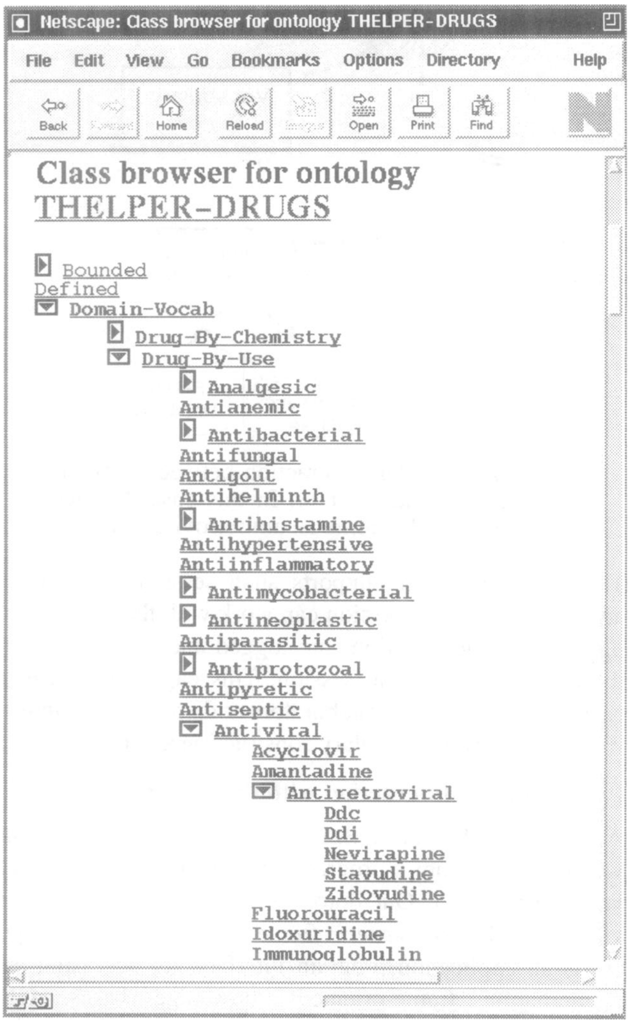
The Ontolingua Knowledge-Representation Language

Just as it is convenient for a programmer to use a language specifically designed for the programming task at hand, the ontology server uses a language specifically designed for representing sharable ontologies: *Ontolingua.* This language is built up from the Knowledge Interchange Format (KIF) [10], a language designed for knowledge sharing with a foundation in formal set theory and logic. Ontolingua is a relatively rich, expressive language, and this design allows Ontolingua to be compatible with a number of different knowledge representation formalisms. The Knowledge Sharing Technology Project currently supports translation from Ontolingua into a number of different languages, including CLIPS, Loom, and the interface definition language (IDL) for CORBA.

Inevitably, legacy and local medical vocabularies will exist at different locations. A medical vocabulary server should therefore include services that assist with *translation* into and out of different vocabulary systems. Ideally, translation capabilities include both the ability to translate across syntax, for example, from Ontolingua to CLIPS, and also the ability to translate the semantics of a medical concept from a legacy vocabulary system into the appropriate term in a shared, common vocabulary. Ultimately, some of the translation costs must be born by the local, legacy systems. However, the vocabulary server should provide support for local translation projects. As a first step, it is important that the server representation language be one that is compatible with a variety of different formalisms. For this reason, we believe that the knowledge-sharing design principles of Ontolingua make it an appropriate formalism for a medical vocabulary server.

Browsing and Editing with the Server

With the ontology server, both browsing and editing capabilities are implemented with standard HTML pages and forms. This provides an important advantage to users: Browsing and editing can be performed in a platform-independent way, using standard web browsers such as Netscape or Mosaic. By making the vocabulary available to a wide set of users, we hope to encourage the type of collaborative development that is essential for the construction of a useful medical vocabulary. Thus, it is imperative that tools for vocabulary editing be platform independent, allowing researchers at different sites with different computing environments to contribute to vocabulary development.



Class browser for ontology THELPER-DRUGS

- Bounded
- Defined
- Domain-Vocab
  - Drug-By-Chemistry
  - Drug-By-Use
    - Analgesic
    - Antianemic
    - Antibacterial
    - Antifungal
    - Antigout
    - Antihelminth
    - Antihistamine
    - Antihypertensive
    - Antiinflammatory
    - Antimycobacterial
    - Antineoplastic
    - Antiparasitic
    - Antiprotozoal
    - Antipyretic
    - Antiseptic
    - Antiviral
      - Acyclovir
      - Amantadine
      - Antiretroviral
        - Ddc
        - Ddi
        - Nevirapine
        - Stavudine
        - Zidovudine
      - Fluorouracil
      - Idoxuridine
      - Immunoglobulin

Figure 2. A small portion of the T-HELPER drug list as stored in the ontology server and viewed with Netscape.

The ontology server supports several ways to browse and search through a vocabulary: Figure 2 shows one example of a browsing view. Each drug name is a link to a page with information about that drug, such as definitions and other attributes that the vocabulary developer may have provided. Additionally, with the same web-browsing tool, users can change and augment information in the vocabulary by filling out and submitting hypertext forms that are generated by the server. For the development of a consensus medical vocabulary, there will be many vocabulary maintainers. Thus, an important capability of the ontology server is the support of distributed, parallel editing sessions.

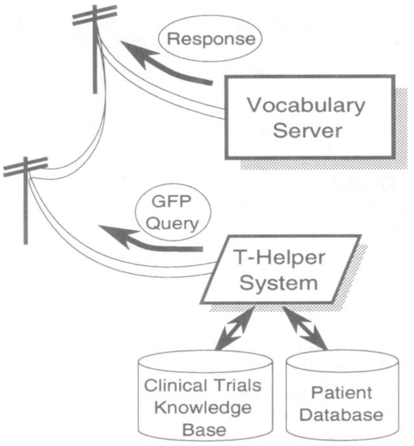


Figure 3. The interaction between T-HELPER and the vocabulary server. In this case, the query might be "Is zidovudine an antiretroviral drug?"

The ontology server supports such sessions: All clients attached to a shared session can work with the same ontology, and all modifications are logged and broadcast to all those connected to that session. This capability makes it easier for health-care collaborators at disparate locations to work together on vocabulary construction and maintenance.

Programmatic Interaction with the Server

As exemplified by the T-HELPER system, many applications would benefit from a programmatic interface to a medical vocabulary server. Our ontology server provides this service using GFP, the *generic frame protocol* [8]. This protocol is an application program interface for accessing and manipulating frame-oriented knowledge bases. GFP includes two types of operators: ones for retrieving information, and ones for modifying the information in the knowledge base. The modifying operators were not needed for the T-HELPER application, but they would be important for any application used for vocabulary maintenance.

As we discussed in Section 2, for T-HELPER to determine the eligibility of a patient for a clinical trial, it needs to know if one drug is a "kind-of" another drug. T-HELPER originally answered this type of question by querying its locally-maintained drug hierarchy. To connect T-HELPER to the vocabulary server, we simply added code that connects to the server, and then applies the appropriate GFP retrieval operator. Figure 3 shows the interaction between T-HELPER and the ontology server. The server can respond to programmatic queries from applications concurrently with vocabulary maintenance activities and the editing and browsing operations described earlier.

The generic frame protocol is a pre-enumerated set of operators. Although this set covers a wide range of queries, it is not sufficient to answer all types of queries efficiently. For example, if the vocabulary includes a hierarchy of diseases, one might want to query for "the names of the diseases that are located in the kidney." To answer this type of query, the system would have to scan all descendants of the "disease" class (presumably a very large number of classes), and return those that have the value "kidney" for the attribute "site-of-disease." Theoretically, this could be answered by using many separate GFP queries, but this would be inefficient compared to sending the entire query to the server as a single transaction. Fortunately, the ontology server also includes the ability to process a more general type of query specified by an arbitrary boolean combination of predicates with variables. Whether the application programmer uses GFP queries or this more general querying functionality, the details of the client–server communication are hidden by the application program interface.

4. DISCUSSION

Many researchers in medical informatics share the goal of building an accessible medical vocabulary server. However, because there are many different uses of such a server, we believe it is important to focus first on a communication protocol and an architecture for collaborative development, rather than on issues of representational and content choices. This approach is different from the work in developing the GALEN vocabulary server [11]. Although we share many of its goals, GALEN includes a pre-defined structure and framework in which all vocabulary development must occur. In contrast, we plan to delay making any representational committments for medical concepts until we have a clear idea of the scope of uses of the medical vocabulary server.

To help us make appropriate choices about vocabulary structure and knowledge representation schemes, we should first identify a set of medical applications that need specific functionalities from the vocabulary server. Medical applications such as the T-HELPER system could be used both to test different types of vocabulary organization and representation and to test different protocols for client–server interaction. By developing and disseminating a test set of diverse medical applications, we can explore a wider variety of vocabulary server capabilities.

In this paper, we have proposed the ontology server as an initial architecture for collaborative development of a medical vocabulary, and a protocol for programmatically interacting with that vocabulary. The key advantage of the ontology server is that it can be used in a platform-independent manner. With the ontology server, collaborative development of a medical vocabulary can occur using standard web-browsing tools such as Netscape or Mosaic. There are no other installation procedures or software or hardware requirements.

The small vocabulary of drug names used by T-HELPER and installed in our current prototype is neither a very rich knowledge representation, nor very complete in comparison to our long-term goals. Thus, the next issues we plan to investigate are choices in representation and structure for medical concepts, as well as issues of scale when the vocabulary grows in size and scope.

Choices in representation will become important as we attempt to provide vocabulary *translation* services. As mentioned earlier, an important service to provide will be the ability to understand legacy vocabularies at different sites. We do not believe that a single, canonical medical vocabulary can ever be built that satisfies the needs of every local site. Instead, a vocabulary server should provide mechanisms that assist with translation across vocabularies, and the ability to download portions of a vocabulary to a particular site. These requirements suggest that choices in representation and structure should make as few constraining commitments as possible, since this will allow easier translation across a wider scope of vocabularies.

As the vocabulary grows in size, there are a number of scaling problems that must be overcome. In addition to basic problems of storage space and response time, the user-interface may need to be redesigned: An interface for vocabulary browsing that works well with smaller vocabularies (less than 1000 concepts) may be awkward and inappropriate for vocabularies that are orders of magnitude larger in size. Also, response time for arbitrary queries could become problematic as the vocabulary grows. Currently, there is little or no effort made to optimize queries. Query response time may or may not be critical depending on whether the medical application needs the vocabulary information at *run-time* (such as the T-HELPER system) or at *design-time*. In the latter case, an application could periodically compile or cache the results of a query to the vocabulary server. For such applications, query response time is less important.

These problems must be faced before we can develop a useful medical vocabulary and a server architecture for providing access to that vocabulary. To date, we have shown the feasibility of the ontology server as a prototype medical vocabulary server. We have demonstrated this capability by using the ontology server with the T-HELPER system and a vocabulary of drugs. More important for our long-term goals, the ontology server provides features essential for collaborative development of a shared resource: (1) Users can edit and browse in a platform-independent way, by using standard web-browsing tools, and (2) the information stored is programmatically available with a general-purpose interface. By making this architecture readily available to the widest audience possible, we hope to encourage collaboration in our effort to construct a medical vocabulary that serves a wide variety of health-care applications and needs.

Acknowledgments

The InterMed project is funded by contract N01-LM-4-3514 from the National Library of Medicine, the T-HELPER project by grant HS06330 from the Agency for Health Care Policy and Research, and the Ontolingua project by the Advanced Research Projects Agency (ARPA order #8607) and NASA/ARC contract NAG 2-581. Jim Cimino, Ted Shortliffe, and Ed Pattison-Gordon, among other InterMed collaborators, helped form our initial ideas. Samson Tu helped with integration into the T-HELPER project. Adam Farquhar and Richard Fikes provided expertise and assistance with Ontolingua. Debby Fife provided editorial assistance.

References

[1] Barnett, G.O., Zielstorff, R.D., Piggins, J., et al. (1982). COSTAR—a comprehensive medical information system for ambulatory care. *Proceedings of the Sixth Annual Symposium on Computer Applications in Medical Care*, pp. 8–18, New York, NY.

[2] Stead, W.W., and Hammond, W.E. (1988). Computer-based medical records: the centerpiece of TMR. *MD Computing*, **5**(5), 48–62.

[3] Pryor, T.A. (1988). The HELP medical record system. *MD Computing,* **5**(5), 22–33.

[4] Lindberg, D.A.B., Humphreys, B.L., and McCray, A.T. (1993). The unified medical language system. *Methods of Information Medicine* **32**(4), 281–291.

[5] Farquhar, A., Fikes, R., Pratt, W. and Rice, J. (1995). *Collaborative Ontology Construction for Information Integration*. Knowledge Systems Lab Technical Report KSL-95-63.

[6] Gruber, T.R. (1993). A translation approach to portable ontology specifications. *Knowledge Acquisition,* **5**, 199–220.

[7] Guarino, N., and Giaretta, P. (1995). Ontologies and knowledge bases: Toward a terminological clarification. In N.J.I. Mars (ed.), *Towards Very Large Knowledge Bases*, IOS Press, pp. 25–32.

[8] Karp, P.D., Myers, K. and Gruber, T. (in press). The generic frame protocol. *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, Montreal, Canada. Also see specification document available at the URL *http://www.ai.sri.com/~gfp/.*

[9] Musen, M.A., Carlson, R.W., Fagan, L.M., Deresinski, S.C., and Shortliffe, E.H. (1992). T-HELPER: automated support for community-based clinical research. *Proceedings of the Sixteenth Annual Symposium on Computer Applications in Medical Care*, pp. 719–723, Baltimore, MD.

[10] Genesereth, M.R., and Fikes, R.E. (1992). *Knowledge Interchange Format, Version 3.0 Reference Manual.* Computer Science Department Technical Report Logic-92-1, Stanford University, CA.

[11] Rector, A.L., Solomon, W.D., Nowlan, W.A., Rush, T.W., Zanstra, P.E., and Claassen, W.M.A. (1995). A terminology server for medical language and medical information systems. *Methods of Information in Medicine,* **34**, 147–157.