Terminology Query Language: A Server Interface for Concept-Oriented Terminology Systems

Michael A. Hogarth[1], Michael Gertz[2], Fredric A. Gorin[3]
[1, 3]Applied Medical Informatics Group, UC Davis Health System, Sacramento, CA
[2]UC Davis Dept. of Computer Science, Davis, CA

ABSTRACT

Designers of medical computing applications increasingly require terminology support for their systems. Yet, terminology systems today lack standard methodologies for providing terminology support. This invariably means increased implementation time and expense for system developers who need to use terminologies in their applications. We introduce Terminology Query Language (TQL), a simple query language interface to server implementations of concept-oriented terminologies. TQL is a declarative, set-based query language built on a generic entity-relationship (E/R) schema. TQL defines a common query-based mechanism for accessing terminology information from one or more terminology servers over a network connection.

INTRODUCTION

There is a growing need to develop and deploy robust, enduring terminology systems in today's healthcare enterprise. Cimino has noted that developing adequate mechanisms for symbolic representation of medical information is one of the greatest challenges in medical computing [1].

A significant amount of informatics research has been conducted in this area resulting in several medical terminology systems [2-7]. Many are evolving toward concept-oriented representation architectures, or "knowledge bases of medical concepts"[8]. The development of terminology servers has accompanied the development of terminology systems [9-13]. However, each provides services through different technologies and interfaces.

In this paper, we introduce Terminology Query Language (TQL), a query language interface enabling simple extraction of terminology information from servers implementing concept-oriented terminology systems. The functionality provided by TQL is commonly required by healthcare applications.

TERMINOLOGY SERVER INTERFACES

Relatively little has been published on the notion of a common interface for terminology servers. The most extensive effort in medical computing remains that of Solbrig and Brinson who proposed a CorbaMed Lexicon Query Services (LQS) specification [14]. LQS is designed to work within a CORBA environment and thus involves the use of an object-oriented distributed architecture.

The issue of functional interfaces has also been explored in the knowledge system domain. Open Knowledge Base Connectivity (OKBC), for example, is a protocol for accessing knowledge bases stored in knowledge representation systems[15]. Generic Frame Protocol (GFP), another interface for frame representation systems[16], has been used to develop a Web-based server for sharing of controlled medical vocabularies [17]. GFP specifies a common knowledge model for frame knowledge representation systems (classes, individuals, slots, and facets). It also specifies a set of operations based on this model (e.g., "find a frame matching a name", "enumerate the slots of a frame", "delete a frame").

Many current concept-oriented medical terminology systems are derived from frame-based designs so it would follow that either GFP or OKBC could be used as interfaces for concept-oriented terminology systems. GFP and OKBC assume a frame-based architecture.

TERMINOLOGY QUERY LANGUAGE

A query language interface may offer a simple and extensible solution to the terminology server interface problem. Many query language interfaces are in existence today. Examples include the well known Structured Query Language (SQL)[18] for relational database

1067—5027/00/$5.00

systems as well as the emerging Object Query Language (OQL)[19] for object databases. Other efforts include Object-Protocol (OPM) Query Language[20], Macromolecular Query Language (MMQL)[21], and XML Query Language (XQL)[22].

At UC Davis, we have undertaken the development of a query language as a means of providing a common interface to servers implementing concept-oriented terminology systems. Terminology Query Language, or TQL, is much like other query languages in that it is set-based and declarative. Declarative languages specify what to get rather than how to get it and are therefore fairly easy to learn and use. The TQL specification is based on a generic entity-relationship (E/R)[23] schema for concept based terminology systems (Figure 1).

This approach allows the data structures and names for terminology-specific data types to be mapped to an abstract set of structures with intuitively familiar names and behaviors. Developers and others embedding terminology functionality into applications need only to know the generic schema and TQL syntax. As long as the underlying concept-oriented terminology system's model can be mapped to the TQL E/R schema, TQL can be used as the interface. Thus far, we have mapped SnomedRT and the NHS Clinical Terms (Read V3) to the TQL schema.

TQL SYNTAX

As noted, TQL uses a generic E/R schema as the underlying information model. The basic syntax of TQL is as follows:

<tql_query> ::= **get** <attrib>+|* **from**
<entity>.<attrib>.<selector> [/, +, - <tql_query>]

<entity> ::= Concepts | Terms | C_Rel | T_Rel
<attrib> ::= entity-specific attribute
<selector> ::= <function(<param>) | All()
<param> ::= letter {letter | digit}
letter ::= A-Z | a-z
digit ::= 0-9

The following set operations are possible:
/ = intersects
+ = union
- = difference

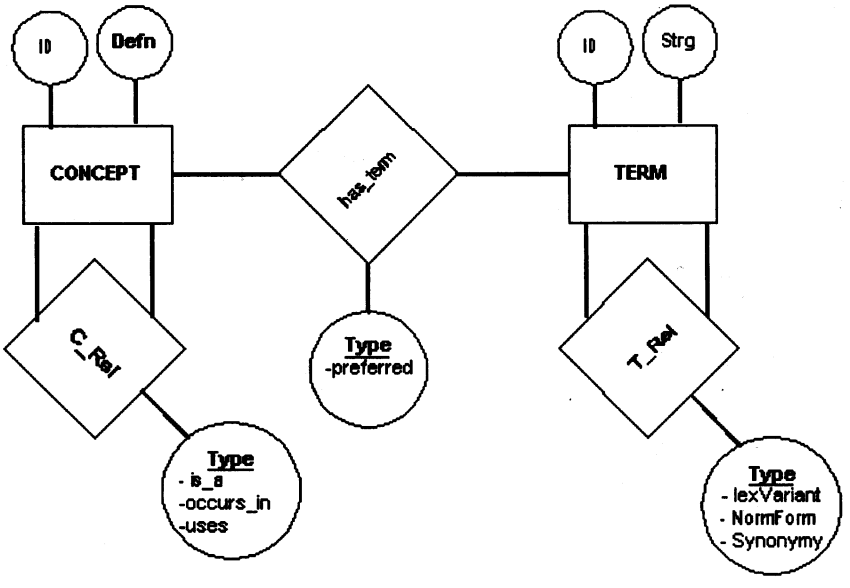

FIGURE 1: The E/R schema used by TQL
(ID = unique ID; Strg = String; Defn = Definition;
C_Rel = concept relationship; T_Rel = term relationship)

Examples of TQL 1.0 Query Types
get ConceptID **from** Concepts.All() {list of ConceptID's for all concepts in the terminology}
get ConceptID **from** Concepts. PreferredTerm.isExact("Asthma") {retrieves unique external identifier for concept that has "Asthma" as a preferred term}
get * **from** Concepts.ConceptID.isExact("D2-51000") {return all attributes for a given concept unique identifier}
get Concept **from** Concepts.RelationshipString.isExact("is_a") {return concepts that have "is_a" as a relationship with other concepts}
get TermID **from** Terms.StringRepresentation.startsWith("Oligocytosis"), **get** Concept **from** Concepts.PreferredTerm.isExact("Anemia") **get** TermID **from** Terms.StringRepresentation.hasPartial("chronic disease") {examples of pattern matching to obtain either Concept or Term attributes}
get String **from** Terms("D2-51000.1").Relationship("Synonymy") {retrieve the synonyms of the term specified by "D2-51000.1"}

TABLE 1. Example TQL queries.

The current TQL specification is limited to information extraction operations. Plans are underway to add terminology system maintenance and update functionality through the use of **put**, **remove** and **modify** constructs.

USING TQL

TQL provides a mechanism for operating on groups of Concept or Terms, traversing the information space defined by a particular concept-to-concept relationship, and extracting attributes for a particular entity in the terminology. An example set of TQL queries is shown in Table 1. TQL output is structured in XML according to the structures in the result set. Items in the result set can only be in the form of one or more of the data structures defined by the TQL E/R schema (Figure 1). XML provides a convenient "transfer" format for handing information back to the system requesting terminology information.

XML output example from a TQL server:

TQL query given to the server
get String **from**
Terms("D2-51000.1").Relationship("Synonymy")

Query semantics:
get synonyms for the term that has Term ID = D2-51000.1 (ie, "Anemia" in SnomedRT).

Output sent back to the client issuing the TQL query:

```
<XML version="1.0"?>
<tql-result-set>
  <tql-query>
        get String from
        Terms("DC-
        10010.1").Relationship("Synonymy")
  </tql-query>
  <term>
      <termid>
        DC-10010.2
      </termid>
      <string>
        Anemia, NOS
      </string>
  </term>
  <term>
      <termid>
        DC-10010.3
      </termid>
      <string>
        Oligocythemia of red blood cells
      </string>
  </term>
  <term>
   ...
  </term>
 ...
</tql-result-set>
</XML>
```

TQL 1.0 can provide several of the desirable terminology server functions detailed by Chute and colleagues [14, 25](Table 2).

CONNECTION STRATEGIES

TQL is protocol neutral. The specification does not define a particular connection strategy. Because of this neutrality, TQL can be used with any number of high-level application protocols

such as HTTP or perhaps CORBA. Additionally, software drivers for particular protocols can be developed for TQL-enabled servers.

Conceivably, TQL drivers could be constructed according to the ODBC or JDBC standard, making available a wide array of transaction management systems in a distributed healthcare computing environment.

Desired Terminology Server Functionality (Solbrig and Chute)	Supported by TQL 1.0
Enumeration of concepts in the terminology	✔
Retrieval of unique external identifier for concept	✔
Retrieval of attribute values which translate to other coding schemes	P
Enumeration of attribute types and relationships in a terminology	✔
Enumeration of concepts which participate in specified relationships	✔
Enumeration of attributes and relationships for a given concept	✔
Enumeration of concepts with a particular attribute value	✔
Enumeration of concepts that satisfy multiple relationships and attribute value combinations	P
Partial pattern matching	✔
Traversal of relationships within a terminology	✔
Representation of concepts as coordinated terms or composite entries	*
Word normalization	P
Word completion	P
Specifying a target terminology	P
Spelling correction	P
Lexical matching	P
Term completion	✔
Semantic locality	✔
Term composition	*
Term decomposition	P

TABLE 2. Functionality provided in TQL
(✔= supported , P= planned, *=not planned)

TQL SERVER

In an effort to explore the usability of TQL, a prototype TQL-enabled terminology server (UC Davis TQL Server) has been developed by the authors. This server provides TQL-based access to the SNOMED RT terminology system (beta 2 version, 2000) and the NHS Clinical Terms (ReadV3, 1999). For demonstration purposes, a Web enabled front-end to the UC Davis TQL server has been developed (http://kenai.ucdavis.edu/termserver; login=user, password = demo).

CONCLUSION

In this paper, we have described a generic model that can be mapped to concept-oriented terminologies and a query language based on this model. We have also shown that from a functional perspective, TQL can provide many of the functions felt to be required of a terminology server.

Ideally, terminology server interfaces should be simple, platform-neutral, use a well-defined model, and be extensible. We believe TQL has these qualities and provides a powerful, yet simple mechanism for developers of concept-oriented terminology systems to expose their system's functionality to application developers.

Acknowledgements:
The authors wish to acknowledge the College of American Pathologists, the UK's NHS, and the National Library of Medicine for graciously providing Snomed RT, NHS Clinical Terms, and the UMLS respectively for this work.

REFERENCES:

1. Cimino JJ. Vocabulary and health care information technology: state of the art. Journal of the American Society for Information Science 1995;46(10):777-82.
2. Read Codes, Version 3. NHS Management Executive, Department of Health, London: 1994 1999.
3. Lindberg DA, Humphreys BL, McCray AT. The Unified Medical Language System. Methods Inf Med 1993;32(4):281-91.

4. Cote RA, Robboy S. Progress in medical information management. Systematized nomenclature of medicine (SNOMED). Jama 1980;243(8):756-62.
5. Spackman KA, Campbell KE, Cote RA. SNOMED RT: a reference terminology for health care. Proc AMIA Annu Fall Symp 1997:640-4.
6. Rector AL, Nowlan WA. The GALEN project. Comput Methods Programs Biomed 1994;45(1-2):75-8.
7. 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, editor.; 1995; 1995. p. 421-5.
8. Cimino JJ. Distributed cognition and knowledge-based controlled medical terminologies. Artificial Intelligence in Medicine 1998;12(2):153-68.
9. Rector AL, Solomon WD, Nowlan WA, Rush TW, Zanstra PE, Claassen WM. A Terminology Server for medical language and medical information systems. Methods Inf Med 1995;34(1-2):147-57.
10. Nowlan WA, Rector AL, Rush TW, Solomon WD. From terminology to terminology services. Proc Annu Symp Comput Appl Med Care 1994:150-4.
11. McCray AT, Razi AM, Bangalore AK, Browne AC, Stavri PZ. The UMLS Knowledge Source Server: a versatile Internet-based research tool. Proc AMIA Annu Fall Symp 1996:164-8.
12. Ontyx. Ontylog Distributed Terminology System (http://www.ontyx.com/products.html).
13. Rocha RA, Huff SM, Haug PJ, Warner HR. Designing a controlled medical vocabulary server: the VOSER project. Comput Biomed Res 1994;27(6):472-507.
14. Solbrig H, Brinson T. Lexicon Query Service: RFP Response. Murray, UT: 3M Health Information Systems; 1998.
15. Chaudhri VK, Farquhar A, Fikes R, Karp PD. Open Knowledge Base Connectivity 2.0.3. Palo Alto, CA: Artificial Intelligence Center SRI International and Knowledge Systems Laboratory Stanford University; 1998 April 9, 1998.
16. Karp PD, Myers K, Gruber T. The generic frame protocol. In: International Joint Conference on Artificial Intelligence; 1995; 1995. p. 768-774.
17. Gennari JH, Oliver DE, Pratt W, Rice J, Musen MA. A web-based architecture for a medical vocabulary server. Proc Annu Symp Comput Appl Med Care 1995:275-9.
18. Date CJ, Darwen H. A Guide to the SQL Standard, 3rd Edition. Reading, MA: Addison-Wesley Publishing Company; 1992.
19. Cattell RGG, Barry DK. The Object Database Standard: ODMG 2.0. San Francisco, CA: Morgan Kaufmann Publishers, Inc.; 1997.
20. Chen I-MA, Kosky A, Markowitz VM, Szeto E. The OPM Query Language and Translator. Berkeley, CA: Lawrence Berkeley National Laboratory; 1996.
21. Shindyalov IN, Chang W, Pu C, Bourne PE. . San Diego, CA: San Diego Supercomputer Center.
22. Robie J, Lapp J, Schach D. XML Query Language (XQL): World Wide Web Consortium; September, 1998.
23. Chen PP-S. The Entity-Relationship Model--Toward a Unified view of Data. San Mateo, CA: Morgan Kaufmann; 1988.
24. Date CJ. An Introduction to Database Systems: Addison-Wesley Publishing Company; 1994.
25. Chute CG, Elkin PL, Sherertz DD, Tuttle MS. Desiderata for a clinical terminology server. Proc AMIA Symp 1999:42-6.