An extensible network query unification system for biological databases

D.Curtis Jamison\textsuperscript{1,3}, Brad Mills\textsuperscript{1} and Bruce Schatz\textsuperscript{2}

Abstract
Database federation enables biological researchers to utilize resources more effectively, creating an environment in which the researcher can query multiple data sources without spending time learning new query mechanisms or issuing redundant queries which need to be integrated. Several mechanisms exist to federate databases. The ENQ\textsuperscript{u}ire system is a network database federation system which uses a World-Wide-Web (WWW) interface to connect the users to various databases. Generic queries entered via a query generator form are sent in parallel to multiple databases, and the results are presented to the user in a unified format. All forms building, query generation, and results translation is done on the fly, and individual database translation modules can be added dynamically. ENQ\textsuperscript{u}ire is a flexible answer to the problems of database federation on the WWW.

Introduction
A revolution is occurring in the biological sciences. Genome sequencing projects are under way for a variety of organisms, including the nematode (Wilson, 1994), yeast (Levy, 1994), and human genomes (Olson, 1993), and the first complete genome (Haemophilus influenzae) (Fleischmann et al., 1995) has recently been published. Sequence data is growing exponentially, and as sequencing projects proceed, the nature of bench work will change, becoming increasingly dependent upon information retrieval.

Currently, biological researchers rely heavily upon sequence databases like GenBank, PIR, Swiss-Prot and PDB which contain sequence information for all species; literature databases like MEDLINE and Biosys; and species-specific genetic databases like GDB, FlyBASE, and ACeDB. While most of these sources are easily accessible by ordinary computers, they are accessed by disparate query mechanisms, forcing the researcher to understand several different query languages and making the integration of results from different databases difficult. The federation of biological databases is an important step toward providing a single biological dataspace.

A large obstacle to database federation is the variety of data models and schemas used by the various databases. Data models range from flat-file formats left over from the days of card decks to the latest in object-oriented technology. Schemas differ greatly both in data representation and in data emphasis. The challenge to database federation becomes one of forging links or combining databases without losing information.

There are two general approaches to database federation. The warehouse approach brings the databases to one location, usually producing a tight federation by producing one or more secondary databases. Examples of the warehouse approach are systems like IGD, ENTREZ and OWL. These are non-redundant federations which transform heterogeneous data models and schemas into a single format (Bleasby and Wooten, 1990; Boguski, 1994; Ritter, 1994).

Alternatively, a virtual federation approach produces a loose federation by creating links between related database entries. Examples of this approach are systems like DBGET and SRS. DBGET relies upon a secondary database of similarity links (Akiyama and Kanehisa, 1995), while SRS creates indices to provide cross-references (Etzold and Argos, 1993).

The Worm Community System (WCS), a community system for \textit{C.elegans} researchers, used the warehouse approach to federate information from ACeDB, literature sources, and informal (uncurated) user submissions (Shoman et al., 1995). WCS was a pioneer federation system, using a very tight federation to produce a uniform user interface and data base. Despite the success of user interface, there were several drawbacks in the data base used to support the front end. The primary problem involved updating the federation, especially as the ACeDB data schema changed through multiple iterations. When the decision was made to migrate from the custom, single-use WCS user interface to a more generic WWW-based interface, a virtual federation using remote data sources was pursued.

\textsuperscript{1}National Center for Supercomputing Applications and
\textsuperscript{2}Community Systems Laboratory, National Center for Supercomputing Applications, University of Illinois, Urbana-Champaign, Urbana, IL 60801, USA
\textsuperscript{3}To whom correspondence should be addressed. Present address: National Agriculture Library, USDA, 10301 Baltimore Blvd., Beltsville, MD 20705, USA. E-mail: jamison@ncsa.uiuc.edu
The resulting technology proved to be a relatively simple answer to database query federation on the WWW. The system was extended beyond the data sources originally found in WCS, and the result is the ENQuire (Extensible Network Query Unifier) system (URL: http://csl.ncsa.uiuc.edu/ENQuire), a scalable HTTP gateway for querying distributed databases. ENQuire provides database query federation at the syntactic level, and creates a browsable dataspace in which databases are linked at the semantic level. The federation is done on the fly, and the method of adding databases into the system is simple enough to allow new databases to be added quickly (two to three days, on average).

**System and methods**

ENQuire is a CGI application written in perl (Wall and Schwartz, 1991). It requires a forms capable WWW browser such as Mosaic or Netscape as a client, and it relies upon translation libraries to translate from a generic query language into database-specific query languages. Translated queries are submitted to the appropriate database, then the responses are collected and collated prior to being interpreted into HTML format and returned to the user. This flow is illustrated in Figure 1.

Perl5 was chosen as the implementation language based on the strength of the string intrinsics, which are better than those found in many languages. Perl5 also offers object-oriented programming. A third factor in favor of perl5 is the interpretive nature of the language which, at a

---

**Table I. Required subroutines**

<table>
<thead>
<tr>
<th>Subroutine</th>
<th>Action</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;DB&gt;_qt</td>
<td>Translates from generic query to database specific query</td>
<td>INPUT: search list, class list, term list, query list</td>
</tr>
<tr>
<td>&lt;DB&gt;_rt</td>
<td>Translates and summarizes results into HTML anchors pointing at object server</td>
<td>INPUT: result list, OUTPUT:</td>
</tr>
<tr>
<td>&lt;DB&gt;_os</td>
<td>Translates a link to a specific database object into a corresponding query</td>
<td>INPUT: class string, term string, query string, OUTPUT:</td>
</tr>
<tr>
<td>&lt;DB&gt;_ot</td>
<td>Translates a single item into perl5 object</td>
<td>INPUT: result string, OUTPUT: HTML string</td>
</tr>
<tr>
<td>&lt;DB&gt;_return_class</td>
<td>Returns a comma separated list of all possible return types</td>
<td>INPUT: none, OUTPUT: class list</td>
</tr>
<tr>
<td>&lt;DB&gt;_query_class</td>
<td>Returns a comma separated list of all possible search classes or types</td>
<td>INPUT: none, OUTPUT: class list</td>
</tr>
<tr>
<td>&lt;DB&gt;_cm</td>
<td>Optional communications module</td>
<td>INPUT: query list, OUTPUT: result string</td>
</tr>
</tbody>
</table>

These subroutines are required to create a translation protocol library. The action performed by the subroutine is indicated, along with the required parameters and the required output. The reference '<DB>' indicates the name of the database, e.g. the ACeDB translator file 'ACeDB_bpl.pl' contains the subroutines 'ACeDB_qt' etc.
Fig. 2. An ENQUire Query Session. The left screen shows the database selection form. Checkboxes are used to select databases. The center screen shows the query generation form. The search types are calculated as the intersection of the search types available from all the databases. The right screen shows the results page, which presents the names of the objects found by the search, grouped by database. Each entry is a hyperlink to the object server program, which creates the query needed to fetch the desired object.

minor speed cost, allows for the dynamic addition of translation modules without the need for recompilation. Finally, perl is platform independent, allowing the gateway to be installed on any UNIX system without recompilation.

Each translation module defines five required subroutines and one optional subroutine. These are briefly defined in Table I. Within the confines of the defined interfaces, any programming approach for the routines is supported. Information regarding the authoring of translation modules can be found at http://csl.ncsa.uiuc.edu/ENQUire.

Implementation

A sample ENQUire query session is shown in Figure 2. A database selection screen is presented to the user (left screen), in which check boxes are selected. The intersections of the databases' return types and query types are computed, and a query-builder form is returned to the user (center screen). The user can build a complex query of up to seven terms, nested by parentheses and linked by either AND or OR. The search results are presented to the user in a unified format (right screen), with the source database and the links clearly defined.

The query federation process illustrated in Figure 2 is at the syntactic (field names) level. Basic biological types are given common names, and the common names are translated into the proper terminology for a specific database. The field name mapping is specific by database, and sometimes specific by position within the database. For example, within ACeDB, a person object is referred to by several field names depending upon the context.
Following links from the results summary will connect the user with pages drawn from the original databases. A sample set of results pages is shown in Figures 3 and 4. In Figure 3, the *daf-4* gene entry from ACeDB has been retrieved and scrolled down to view the references. Each link (underlined text) in the entry points at an object server specific for that database, which creates the proper query to retrieve and translate objects from that database. The object server can redirect requests, retrieving an object from a more canonical source, creating a semantic (field value) level federation.

Figure 4 shows the result of retrieving a *Worm Breeder’s Gazette* article from the ACeDB object server. A redirection has occurred, and the HTML page from the WBG archives at NCSA has been retrieved, as opposed to the normal ACeDB object for the same item, shown in Figure 5. The ACeDB version of the text is not hyperlinked, and is presented as a database record rather than as a document. The WBG documents in the NCSA archive are a better presentation of the data. While this

![Fig. 3. ACeDB Entry. The reference portion of the ACeDB entry for the gene *daf-4*, as retrieved by ENQUire from the WWW ACeDB server at USDA/NAL. The server delivers a perl5 object, which is translated into HTML with the links pointing at the ENQUire object server script](image1)

![Fig. 4. HTML Version of a WBG Article. The object server script for ACeDB redirected the WBG article request to the HTML version of the WBG at NCSA. The internal links in the hypertext document point at the ACeDB object server, since all the links refer to *C. elegans* genes.](image2)

![Fig. 5. ACeDB Version of a WBG Article. These two screens illustrate the standard ACeDB version of WBG articles and other text documents. The first page shows bibliographic data, and the next shows the plain ASCII text without any hyperlinks.](image3)
example is somewhat superficial, the same simple semantic federation can be used to follow sequence accession numbers between literature and sequence databases. The net result of redirecting the fetches is an apparent increase in the dataspace seen by the user.

The databases available in the initial release of ENQUire reflect both the WCS roots of ENQUire and a sampling of important molecular biology databases. Five databases are available: ACeDB and the WBG archive (from the WCS roots), as well as SolGenes, Entrez, and PIR.

ACeDB and SolGenes are organism specific genome databases based upon ACEDB (Durbin and Meig, 1991). ACeDB contains information about the nematode C.elegans, while SolGenes contains information about the genus Solanaceae (potatoes, tomatoes, and peppers). The databases are queried using the perl module Aceclient.pm, accessing the aceserver maintained at the National Agriculture Library, and results are formatted into HTML using the perl module AceWWW.pm (Barnett and Bigwood, 1995).

Entrez federates several biological sequence databases and a literature reference database (Boguski, 1994). The sequence portion of Entrez contains sequences of multiple organisms from Genbank, EMBL, Swiss-Prot, while the references portion consists of the molecular sequence data subset of Medline. Entrez is accessed using ncelver, a command line interface to the network Entrez server (Rioux et al., 1994).

PIR (Protein Information Resource) is a collection of macromolecular sequences. PIR is accessed via the NBRF program ATLAS.

The WBG archive is a full-text and figures hypertext version of the Worm Breeder’s Gazette, and informal newsletter for C.elegans researchers published by the Caenorhabditis Genetics Center. The articles are stored locally at NCSA, and are indexed and searched via a Glimpse server (Manber and Wu, 1994).

Several of these data sources do not support complex queries. The ACEDB-based databases and PIR support operations upon a single keyset. To allow the user to create complex queries involving boolean operations on multiple sets, each atom (i.e. each target-field-term combination) can be queried separately, and the results are combined logically by the ENQUire server.

Performance of the ENQUire system is acceptable. The time required to return a result was compared against the general WWW ACeDB server at USDA/NAL and the WWW Entrez browser at NCBI. PIR and WBG have no corresponding WWW servers. The ACeDB WWW server averaged 6 s for response to an author query, while ENQUire using the Aceclient.pm mechanism returned results in under two seconds. ENQUire queries using ncelver to query Entrez are slower than using the Entrez WWW page directly, but this is probably due to very different mechanisms for performing the query. In both cases, performing a complex query is a multi-step process, and thus is not directly comparable.

A prototype of the ENQUire system was made available to C.elegans researchers in early May of 1995. The response from the community was enthusiastic. In the first 4 months following the announcement, 387 search queries were posted, representing 128 sites. This represents researchers using the system, and does include curiosity inquiries.

The ENQUire system has been designed as a portable gateway, requiring only a forms-capable HTTP server and a Perl5 interpreter. Each of the communications modules is configurable to point at any source, allowing for installation at mirror sites. The gateway code and installation documentation is available by anonymous ftp from csl.ncsa.uiuc.edu in the /pub/ENQUire/Gateway directory. Documentation for authoring translation libraries is available both on the WWW at URL http://csl.ncsa.uiuc.edu/ENQUire and by anonymous ftp from csl.ncsa.uiuc.edu in the /pub/ENQUire directory.

Discussion

The goal of this research has been to create a virtual federation of biological database resources across the Internet. The ENQUire system has been demonstrated to be a successful query integration mechanism for heterogeneous databases. A single query can gather results from multiple data sources via disparate communication protocols, and then present the results back to the user. The speed of making individual queries is not significantly slower than querying the native database, and since the ENQUire queries are done in parallel there is a great time savings when querying multiple databases.

The ENQUire system is fully scalable and distributed. Additions to the database translators is simple, in most cases taking just a few days to add a database to the system. The only requirement for new databases is that some method of remote access exist. The databases do not need to be mirrored in any way, although it is anticipated that some of the major databases will be mirrored locally. Addition of databases is a facile operation, requiring only a few days to write a new translator from scratch, or a few hours to adapt an extant translator for a similar database. The ease of database addition distinguishes the ENQUire system from other database federation systems.

The ENQUire system is straightforward, translating from a generic query language into database specific query languages, then translating the results into a generic HTML format. The simplicity of the approach does have some inherent limitations. One limitation (inherited from the databases themselves) is found when querying
databases with a browsing language (like ACEDB). While the object-oriented ACEDB browsing language is well suited to an interactive session, to perform a single query requires building a string of browser commands to follow non-reciprocal links between objects, making a full search implementation difficult. To illustrate this using the ACeDB model, one can find the person who performed a genetic experiment by following a direct link from a genetic cross data object to a person object (find two_factor ‘x’; follow mapper), but the reverse search to find all the genetic experiments performed by a person doesn’t work, since the reverse link from a person to their associated genetic mapping data doesn’t exist.

Another limitation in the ENQUire system (and database federation in general) is the diversity of nomenclature between organisms. Since gene names differ between species, searching the Solanaceae database for a specific Caenorhabditis gene name is non-productive, even though both databases contain genes which are functionally equivalent. While this is a biological semantics problem and is out of the current scope of research, it does represent a significant hurdle to query federation systems. However, the query mechanisms of the ENQUire system is flexible enough that when research projects like the Digital Library Initiative (Schatz et al., in preparation) achieve a true semantic database federation (i.e. federation of concepts rather than characters) the methodology can be adapted into the core framework.

The ENQUire system was conceived as a data browsing system. This means that in some cases detailed information cannot be accessed directly via the initial query. This is highlighted by the limited number of potential return types. In some cases, as with ACEDB, the return classes were dictated by the mechanism of making queries. In other cases, the limitation comes from the specific type of database represented, e.g. the limitation of the WBG database to returning only documents, while being searchable by any number of criteria. For all the databases, the native query/browsing language is richer in context than the generic one allowed in the ENQUire system. This suggests that the proper use of ENQUire is as a data discovery tool, to look up general queries and then to refer to the specific database(s) for detailed information.

The ENQUire system is a model for distributed query federation environments. Future expansion of the system will include new database translators, the addition of a data cache and the introduction of a session concept, allowing the user to retain and reference information from multiple searches. The data cache will hold information in a standardized data interchange format, allowing subsections of the data to be easily extracted.

Another potential for this work is the addition of translation libraries to handle submission of results to programs such as multiple alignment, parsimony, and structural analysis programs. The fusion of analysis tools to a distributed query system will result in a WWW-based analysis environment, wherein the researcher can retrieve data from many heterogeneous databases, view it, and manipulate it, using either their own local computers or computational resources at NCSDA and other large-scale computing centers.

Acknowledgements

This work was supported in part by NSF grant BIR19844 from the Database Activities program in the Division of Biological Instrumentation and Resources. DCJ is supported by a NSF CDA-CISE postdoctoral fellowship. The authors wish to thank Doug Bigwood (USDA-NAL) and David George (NBRF) for their cooperation in accessing databases, and Shankar Subramaniam (NCSDA) for his critical reading of the manuscript.

References


Wilson,R. et al. (1994) 2.2 MB of contiguous nucleotide sequence from chromosome II of C.elegans. Nature, 368, 32–38


Received on November 16, 1995, accepted on January 8, 1996