EJB Electronic Journal of Biotechnology ISSN: 0717-3458
Vol.5 No.2, Issue of August 15, 2002

Received April 17, 2002 / Accepted August 7, 2002
RESEARCH ARTICLE

# Database integration with the Web for biologists to share data and information

**Yulu Xia** *
National Science Fundation Center for Integrated Pest Management
Department of Entomology
Department of Computer Science
North Carolina State University
Raleigh, North Carolina, USA
Tel: 919 513 1432
Fax: 919 513 1114
http://cipm.ncsu.edu
yulu_xia@ncsu.edu

**Roland E. Stinner**
National Science Fundation Center for Integrated Pest Management
Department of Entomology
Department of Computer Science
North Carolina State University
Raleigh, North Carolina 27695-7553, USA
Tel: 919 513 1432
Fax: 919 513 1114

**Ping-Chu Chu**
Department of Mathematics and Computer Science
Fayetteville State University
Fayetteville, North Carolina 28301-4298, USA
Tel: 910 672 1070
Fax: 910 672 1070
E-mail: Ping.Chu-Chu@uncfsu.edu

**Keywords:** biological data, DBMS, integration, relational database, programming, standardisation.

**The Internet has fundamentally changed the way we conduct our business. Even 10 years ago or so, it was not possible to order a book or access a sequence database in another continent using the computer in our office. All these become possible due to the progress in information technology. It is especially important for biologists to realise that information technology has brought us enormous opportunities in sharing our data. In this paper, authors review and introduce the technologies relating to biological data, database, dynamic integration of databases with the World Wide Web (Web hereafter), and data standardisation.**

## Biological data

Biological sciences are data-intensive. Many disciplines involve either large-scale field investigations or extensive laboratory analyses. A large amount of data is accumulated from these activities. For example, current genome-sequencing projects generate an enormous amount of data related to the function and the structure of biological molecules and sequences (Setubal and Meidanis, 1997). Other complementary high-throughput technologies, such as DNA micro-arrays, pour out mountains of data in minutes (Adams et al. 1994). There are over 1 million known species of insects found on the planet so far, and the numbers are increasing each day (Earth Llife Web, 2002). Many of these species are economically and/or environmentally important. Consequently, there are substantial amounts of data and information on these creatures. The amount of biological data increases exponentially due to the availability of high efficiency laboratory equipment and technology. The avalanche of incoming data requires us to find efficient ways to store, manage, use, and share these data.

Many biologists use spreadsheets (such as Excel and Lotus) for data storage and analysis. Spreadsheets are easy to learn and convenient to use in many situations. Spreadsheet is also a powerful tool. It can organise data by sorting and filtering lists, account and check totals, calculate complex formula, present the data in graphical form, and support presenting information in a structured layout. Newer versions have incorporated many features that used to be found only in more sophisticated software such as database

* Corresponding author

management systems (DBMS).

In spite of these merits, a spreadsheet lacks many features for storage and management of large amount of data, especially, for sharing data through the Web. A spreadsheet is primarily used and maintained by a single person (computer). This is a limitation in today's scientific environments where cooperation and sharing (search and use) data is a key for success. Newer versions of spreadsheets can be put in a networked computer and shared with other people. However, the functionality and data security of spreadsheet are very limited compared to DBMS (Gordon, 1998). It is either impossible or error-prone to query, search, or upgrade spreadsheets remotely. The work can become even more challenging if the query, search or upgrading is across multiple spreadsheets. Under these situations, a database is a better choice, especially when it is intended to share data across network and/or integrate multiple data sets.

## Database introduction

Databases have been used extensively in certain fields of biology, such as bioinformatics and genomics. In a non-technical term, a database can be defined as a collection of data. The flat file database, which was very common in early database history, is an example of this type of database (Elmasri and Navathe, 1994). The flat file database is mainly used as data storage. It has few features for managing data. Documents in a flat file database are usually text files, and files are independent. Flat file database can be searchable using text search. However, it is almost impossible to do more complicated operations, such as grouping data from several databases together and updating data automatically. Additionally, to develop an application program working on a file, the structure of files has to be known precisely (Bellahsene and Ripoche, 2001). Due to these drawbacks, flat file database is rarely used for any large-scale application. However, flat file databases are widely used in genetic sequence databases like GenBank maintained by National Center for Biotechnology Information (NCBI) (http://www.ncbi.nlm.nih.gov/) (Pittsburgh Supercomputing Center, 1999; Bellahsene and Ripoche, 2001). One of reasons for this is that flat files can be easily distributed and readily comprehensible. Flat files can be transformed and converted to other data format (Jungfer et at. 1999). A list of problems associated with flat files can be found at EMBL's web site (http://corba.ebi.ac.uk/background.html).

Modern database can be described as a collection of data managed by a DBMS (Elmasri and Navathe, 1994). A DBMS is a software system for creating, manipulating, and managing data. Some well-known DBMSs include Oracle, Sybase, DB2, and SQL Server. Regardless of size of a database, all data (either in tables, objects, or hybrid formats as we will discuss later) in a database are associated together by certain relationships or associations depending on database type. This is one of the important features that make a database powerful and efficient. Database allows to store, retrieve, or modify data easily and efficiently regardless of the amount of data being manipulated. What the data are and how demanding you will be when retrieving and modifying that data is simply a matter of scale. Another outstanding feature is that database usually uses a query language to interact with the data in it. Query language is like simplified English language with very limited vocabulary. Previous programming background is not required to learn it. Example will be given later to show how easy it is. There are variations in query languages in each DBMS and vocabulary of each query language might be a little different. The important thing here is that query languages can query many tables in a database or even across databases. Consequently, a large amount of data can be worked together at once. Data from different databases can be incorporated, combined, and searched together.

One of the major benefits in using databases for data storage is for data sharing. This is especially true for biological data due to its complexity, hierarchy, heterogeneity, and dynamics (Moore, 2002). Regardless of type of database, once it is registered with database connectivity tool and has an application program, anyone in anywhere can work on the database without having to understand the underlying database type and/or data format. There are two widely used database connection technologies: Open Database Connectivity (ODBC) from Microsoft and Java Database Connectivity (JDBC) from Sun Microsystems. ODBC and JDBC help to establish a connection with a data source, send queries and update statements to the data source, and process the results.

## A brief on database development

Depending on the project, development of a database can be a major effort or just a simple task. For example, a commercial genomic database can cost millions and take a team years to make it functional. On the other hand, one skilful developer can develop a small-size regional biodiversity database within a month assuming field data is available.

## Two database models

Following two database models are most commonly used in most database projects:

## Relational database model

A relational database is made of tables; each table has columns called fields and rows called records. Tables in a

database are linked together by a pair of primary/foreign keys. The primary key is one or more fields that can uniquely identify each record in a table, and the foreign key is a field that is the primary key of another table. Relational databases use Structural Query Language (SQL) for data creation and manipulation. SQL is the only means by which you can access your database.

### Object-oriented (object) database model

An object database is made of objects and classes. Objects are used to model real-world entities. A class is used to define a group of similar objects with the same data structure and the same operation. Classes are organised in a class hierarchy. Object database uses either Object Query Language (OQL) or object identity (OID) to access the database (Chaudhri and Zicari, 2001; Object Database Management Group, 2002).

Relational database is the dominant model of the two types. Over 90% of market is either a relational database management systems (RDBMS) or primarily RDBMS. This is due its simplicity, technological maturity, broad software support, and the flexibility for future modifications. However, the relational database model has certain major drawbacks, especially when used for biological data. For example, it is very difficult to support structured data. So, it is not suitable to use it to represent structure relationship data such as a biological system tree. It is also hard to use a relational database to handle sequential information such as DNA sequence. Object databases might be the choice for this kind of complex data structure. There is also a tendency for a DBMS to support both relational and object or hybrid database.

Due to the broad differences in designing and implementing a database between the two models, following discussions are about relational databases only.

### Database development

One of the first things in the early stage of database development is to choose a suitable DBMS. There is no a general rule here.

Following discussion is based on author's experience and knowledge.

If the size of database will easily exceed 1 gigabyte and there are many users ( > 10) accessing database concurrently, one of the commercial DBMSs mentioned above may be needed. The cost of a DBMS runs from a little more than $1000 to well over $10,000 depending on your licensing situation. There are also open source DBMSs, such as MySQL (http://www.mysql.com), available for free download. It takes time to get familiar with a DBMS. If you are not so sure about the project and/or just want to get a feeling or experience about database, you may consider using Microsoft Access (Access hereafter) to get started. This DBMS is very easy to use (but is less powerful and lacks of many features for data security and integrity control). It is inexpensive and comes as part of the Microsoft Office suite. An average computer user should be able to set up his/her own database in a short time.

Before embarking any database project, it is required to spend time to learn some basic principles about database design. The quality and efficiency of database is largely decided by the quality of database design. Some basic design techniques, such as entity relation modelling and database normalisation, should always be used regardless the size of database. Detailed discussion of the subject is beyond the scope of this paper. There are many online tutorial sites such as Sol, 1998, or refer to Elmasri and Navathe, 1994 or other references for the matter. Additionally, each DBMS usually contains a detailed tutorial to help get started.

Relational database uses SQL to "talk" to data in database as mentioned earlier. Here is a simple example to illustrate how straight forward SQL is. Assume that a database contains information about plants and their pests. One of the tables in the database is called pestList containing the names of plants and their pests. Now assuming that we want to find all of the diseases in corn, following SQL statement could be used:

```
SELECT diseaseNames
FROM pestList
WHERE cropNames = corn
```

The capitalised words are SQL vocabulary. diseaseNames and cropNames are field (column) names in the table pestList. SQL looks like English!

Of course, the simple task above does not really show the power of a database. A database comes to prime time when task gets more challenging. For example, assuming that we are not only interested in all disease names in corn, but also want to get other information, such as syndromes, characteristics, and control methods associated with each disease. This task can be easily accomplished by issuing a little more complicated SQL statement to query several tables in the database, even across databases, if needed.

Besides retrieval of data from a database, SQL can be used to update, delete, and add data to database conveniently. For example, we can issue a single SQL statement to change all names of a gene in a database regardless of how many items needed to be changed in the database. Otherwise, you would need to go line by line or column by

column to change each record in the data set.

More and more organisations use a collective software system called Laboratory Information Management System (LIMS) or Laboratory Database System (LDS). LIMS is basically an integrated, centralised information management system to handle all common data tasks in a biological lab, such as keep track of experiments, data storage and management, analysis and report, data transfer, data security, and many others (Meisenholder, 1999a; Meisenholder, 1999b). LIMS can also collect and manage the data generated from experiments and analyses, and provide suitable means for laboratory personnel to monitor and control the process (Goodman et al. 1999). The heart of this system might be a DBMS (relational, object, or hybrid) we mentioned above. A LIMS usually has a user-friendly interface that allows you to manage your data concurrently and conveniently.

## Integrate database with the Web

Another key reason using database for data storage is that database can be easily integrated with the Web. This is a real revolution in terms of information sharing and exchanging. It brings us enormous opportunity and convenience for sharing biological data. The Web is a client-server type of network, using a series of protocols collectively called TCP/IP (Transmission Control Protocol/Internet Protocol), for communication (Comer, 2000). Client is any computer or program requesting a service or file. Server is the computer or program accepting the request and returning the requested file. We can use client-server architecture to link our biological database with any users in any place. This makes it possible that we can sit in office querying the sequence data in GenBank, or conducting literature search using the computer instead of searching the index page of a journal. There are several online sites for searching life science related articles, such as HighWire (http://highwire.stanford.edu), BioMedNet (http://www.bmn.com), and e-journal (http://www.e-journals.org).

## Techniques for integrating a database with the Web

Programming technologies are the tools that integrate a database with the Web. Many technologies, such as CGI, ASP, JSP, Coldfusion, and PHP, can accomplish the task. CGI, Common Gateway Interface, used to be the most common technology for the task. It has become less and less popular these days due to its slow execution, slow programming, and declining technical support. Many programming languages, such as Perl, C, C++, can be used in CGI. PHP is an open source technology. This means that it might be free to get started. ColdFusion is a proprietary technology owned by Macromedia (the mother company of Allair, the "father" of ColdFusion). It uses ColdFusion Markup Language (CFML) for programming. ASP and JSP stand for Active Server Page and Java Server Page, respectively. ASP is a Microsoft technology using VisualBasic or other languages for programming. It is primarily used for Windows. JSP uses Sun Microsystems's Java technology. It can run on any computer platform (Unix, Linux, Windows, Mac). For the readers who have some programming background, please refer to the web site (Ebiz-Intellect, 2002) for a comparison of these technologies.

## Process of integrating a database with the Web

In order to make a database Web accessible, software called Web server is needed (it may be within your computer already if you are using a server type of computer like Windows 2000 Server). Web servers can be free, such as Apache from the Apache Software Foundation (http://www.apache.org), or commercial products such as listed at http://serverwatch.internet.com/webservers.html. The main function of Web server is to handle the HTTP (HyperText Transfer Protocol) request and response. It is required for providing production services (for example, a database service in our case) to clients through the Web. Another type of software called application server may be also required. Application server processes the server side programs using the technologies we mentioned above. When a user wants to search a database, he/she first logs into the Web site. The user then selects or types his/her search and sends the request to where server program is. The server program then "talks" to the database using SQL through ODBC and/or JDBC, obtains and formats the output, and sends it back to the user (Figure 1).

Choosing the right programming technology is crucial for the success of projects, at least, for the first one. One technology may require a longer learning curve than another. Some may be more suitable for handling complicated tasks.

Regardless of the technology you choose, the basic programming process follows the logic in Figure 1 too. A HTML Web page lets users enter or select his/her choice. Once users press the "Search" button or similar button, the sever side program is invoked. The server side program first takes what user entered, calls the SQL statement built into the program, and "talks" to the database. After receiving a response, the server side program formats the content, and sends it back to client's browser for display or saves into the designated directory.

Server side program is the centrepiece of whole process. It can be just one file with a few line codes on it, or a collection of hundreds of files. Therefore, cost and duration for developing a project can vary substantially depending

on each project. Server side programming usually requires more programming experience than development of a database. Having an experienced developer is invaluable for the first project. Once someone helps grasp the whole picture, get the first one running, it is much easier to start on another project.

## Data standardisation

One of the wonderful things about the Internet is that user can exchange information with each other without worry about underneath details, such as what product the receiver might use, what data format the product accepts. The recipient will receive an e-mail regardless his/her e-mail software or computer platform. This is because all of software (the software involved in the whole process) implement a series of protocols the standards specifying the things such as how to format data during the transfer, and how to handle error if it occurs. All e-mail software implement SMTP (Simple Message Transfer Protocol) which is one of protocols in TCP/IP we mentioned earlier. The same is true for Web pages because the software on both sides (client and server) implement HTTP which is on the Application layer of TCP/IP protocol layers. Standardisation and openness are the heart and soul of the Internet. Without them, there would be numerous proprietary networks instead of a single open Internet.

Integration of a biological database with the Web provides universal access to database. To achieve universal data sharing, we need standardisation. Standardisation here implies at least two issues: 1) standard terminology and nomenclature (White et al. 2001), and 2) standard format for data submission and storage, exchange, and query. The need for standardisation of biological terminology is obvious. For example, it is not uncommon that there are multiple names for one gene in different databases (Anthony, 2002). How can you provide useful and correct data sharing under such a situation? Many standardisations have been developed or proposed in biology (Frondorf et al. 1997; Michener, 1998; Biodiversity Convention Office Canada, 2001; International Working Group on Taxonomic Databases, 2001; White et al. 2001). Due to the huge amount of data in genomics, it is almost impossible to provide timely and meaningful data sharing without a standard format for data submission and storage. Progress has made in the field in genomic/bioinformatics data (NCBI, 2000).

Exponentially increasing in biological data requires automated data exchange, process, and publishing. One of the prerequisites for automatic data exchange and presentation is standardisation. Standardisation allows a computer to “read” data, and process the data according to the meanings. However, most current standards for biological data communication are mainly designed for human understanding, not for computer. This creates bottlenecks in data sharing and exchange (Simon, 2001). For example, we standardise species name by using a universal scientific name for a species. Regardless of language we speak, or how many common names exist for that species, we understand what species it is once the scientific name is there. Now we want computer to do the exactly same thing—“understand” meaning of your data and process (present the data in a browser or put into another database based on the contents) the data automatically.

One of the technologies for solving this problem isn eXtensible Markup Language (XML). XML is similar to HTML in terms of language origin and format. However, the two have some major differences in terms of application and programming. 1) HTML is for data or document presentation in a browser, XML is for describing data for various uses, including presentation in a browser. 2) HTML can function without help from other technologies. For example, your browser can present a HTML document without knowing corresponding style sheet. XML has to be combined with other technologies to function. 3) HTML uses predefined tags for coding. XML does not have predefined tags, you define your own tags!

You use XML to describe your data in a way similar to HTML. For example, <cornColorGene>the color gene name</cornColorGene> describes the words "the color gene name" between the tag as "a corn color gene". Now assuming that we have a XML document with over 20 million line codes like that describing all genes in corn, we can achieve many things with the same XML document: 1) The application program in company’s disease research group retrieves the document, finds all genes related to corn disease, puts each gene into your relational database according to gene name. 2) The application program in the yield research group retrieves the same XML document, finds all genes related to yield, and puts these into their database. 3) Company's web application group also retrieves the same document, and presents all genes in browser, based on gene group. 4) A university researcher’s application program compares the genes in the XML document with the corn gene database he has, and picks up the genes it does not have only. XML can do much more than the four cases above.

In order to achieve these functionalities, the first thing we need is either a DTD (Document Type Definition) or Schema. Both DTD and Schema are designed to do the same thing: 1) define the legal building blocks of an XML document; and 2) define the document structure with a list of legal elements and the rules (such as required or optional, string type or a number) associated with each element. DTD was dominant before. However, more and more people are using a Schema instead of DTD because

Schema is more power and flexible.

Standardisation is crucial--DTD or Schema is usually agreed by all parties involved, usually industry wide. Due to the enormous benefits of using XML for exchange of business information and data, progress in the field is extraordinarily fast during recent years. This is also true in life science-related industries. You can find the Chemical Markup Language (CML) at http://www.xml-cml.org/, the Bioinformatic Sequence Markup Language (BSML) at http://www.oasis-open.org/cover/bsml.html, and the Gene Expression Markup Language (GEML) at http://www.rosettabio.com/products/conductor/geml/default.htm. Almost all major industries in the U.S. have either their XML standards (a DTD or Schema for XML documents in the industry) or are working in the direction. Once you have a DTD or Schema with the XML document, you can use an XML Stylesheet Language Transformation (XSLT) to transform the XML document into either HTML or XHTML for display in browser. But the key difference with HTML is that one XML document can be displayed in many different ways.

However, the power of XML is mainly in data exchange as we illustrated. The software for process XML document can be either commercially available or developed in house using a programming language (Java is recommended) and either SAX (Simple API for XML) or DOM (Document Object Model). All XML documents based on the specific industry standard (like GEML) can be processed automatically. There are mountains of web sites and books on the subject. The World Wide Web Consortium (http://www.w3.org) is the authorative organisation for XML related standards. There are excellent tutorials on each subject we discussed on the site. For processing an XML document using Java, SAX, DOM, XSLT, please refer to Sun MicroSystems's site (http://java.sun.com/xml/tutorial_intro.html) for details.

A series of new technologies based on XML are being explored extensively. These technologies will fundamentally change the landscape of data and information communication (Port, 2002). SOAP (Simple Object Access Protocol), which is based on XML format, is the standard protocol for message exchanging in Web Services (http://www.ws-i.org) and MicroSoft's .NET architecture. Together with newer programming technologies, XML will help us solve the bottleneck in biological data communication and integration we discussed above. This will speed up our research and development in drug and other biological product development.

## References

ADAMS, Mark D.; FIELDS, Chris J. and VENTER Craig. *Automated DNA sequencing and analysis*. London, Academic Press, 1994. 368 p. ISBN 0127170103.

ANTHONY, Leslie. *The quest for data integration*. Incyte Genomics, Inc. [on line]. 2002 . [cited 23 April 2002]. Available from Internet: http://www.incyte.com/insidegenomics/.

BELLAHSENE, Zohra and RIPOCHE, Hugues. An object-oriented databases for managing genetic sequences. In: CHAUDHRI, Akmal B., and ZICARI, Roberto, eds. *Succeeding with Object Databases: A practical look at today's implementations with Java and XML*. New York, John Wiley & Sons, 2001, p. 343 - 356. ISBN 0471383848.

Biodiversity Convention Office, Environment Canada. *The United Nations Convention on Biological Diversity: Clearing House Mechanism and Interoperability of National Nodes* [online]. 2001 [cited 12 May 2002]. Available from Internet: http://www.biodiv.org/doc/ref/chm-interoperability-en.doc.

CHAUDHRI, Akmal B. and ZICARI, Roberto. *Succeeding with Object Databases: A practical look at today's implementations with Java and XML*. New York, John Wiley & Sons, 2001. 442 p. ISBN 0471383848.

COMER, Douglas E. *Internet working with TCP/IP, Principles, Protocols, and Architectures*. Upper Saddle River, New Jersey, Prentice Hall, 2000. 750 p. ISBN 0130183806.

Earth Life Web. *Insects* [online]. 2002 [cited 20 April 2002]. Available from Internet: http://www.earthlife.net/insects/six.html.

Ebiz-Intellect. *ASP, JSP, PHP, CFML? - Compare the most popular server-side scripting languages* [online]. 2002 [cited 12 May 2002]. Available from Internet: http://www.ebiz-intellect.com/page.cfm?onumber=36.

ELMASRI, Ramez and NAVATHE, Shamkant B. *Fundamentals of database*. 2nd ed. Addison-Wesley Pub. Co., 1994. 873 p. ISBN 0805317481.

FRONDORF, Anne; NYQUIST, Maurice and WAGGONER, Gary. *Standards to support exchange of biological data through the National Biological Information Infrastructure* [online]. 1997. [cited 2 May 2002]. Available from Internet: http://www.nbii.gov/about/pubs/presentation/codata.html.

GOODMAN, Nathan; ROZEN, Steve and STEIN, Lincoln. LABBASE: data and workflow management for large scale biological research. In: LETOVSKY, Stanley ed. *Bioinformatics: databases and systems*. Boston, Kluwer

Academic Publishers, 1999, p. 279-292. ISBN 079238573x.

GORDON, Kurtiss. Spreadsheet vs. Database: Which makes more sense?. *UMass Office of Information Technologies* [online]. 1998 [cited 6 March 2002]. Available from Internet: http://www.oit.umass.edu/publications/at_oit/Archive/fall98/spread.html.

International Working Group on Taxonomic Databases. International Union for Biological Sciences Taxonomic Database Working Group. *TDWG STANDARDS* [online]. 2001 [cited 10 March 2002]. Available from Internet: http://www.tdwg.org/standrds.html.

JUNGFER, Kim; CAMERON, Graham and FLORES, Tomas. EBI: CORBA and the EBI Database. In: LETOVSKY, Stanley ed. *Bioinformatics: databases and systems*. Boston, Kluwer Academic Publishers, 1999, p. 279-292. ISBN 079238573x.

MEISENHOLDER, Grant. Out on a LIM: database system for biotechnology. *The Scientist* [online]. 1999a [cited 23 April 2002]. Available from Internet: http://www.the-scientist.com/yr1999/sept/profile2_990927.html.

MEISENHOLDER, Grant. The Paperless lab: database systems for the life sciences , *The Scientist* [online]. 1999b [cited 23 April 2002]. Available from Internet: http://www.the-scientist.com/yr1999/sept/profile3_990927.html.

MICHENER, William K. Ecological Metadata. *Long Term Ecological Research* [online]. 1998 [cited 25 April 2002]. Available from Internet: http://lternet.edu/documents/data-informationmanagement/DIMES/html/michener.fv2.htm.

MOORE, Samuel K. Harmony your data. *Institute of Electrical and Electronics Engineers, Inc., IEEE Spectrum.* [online]. 2002 [cited 25 April 2002]. Available from Internet: http://www.spectrum.ieee.org/WEBONLY/publicfeature/jan01/1_4a_geno.html.

National Center for Biotechnology Information (NCBI). *GenBank Annotation Examples*. NCBI [online]. 2000 [cited 25 April 2002]. Available from Internet: http://www3.ncbi.nlm.nih.gov/BankIt/examples/requirements.html.

Object Data Management Group. *Standard Overview* [online]. 2002 [cited 5 April 2002]. Available from Internet: http://www.odmg.org/standard/standardoverview.htm.

Pittsburgh Supercomputing Center. *Using the GenBank Database* [online]. 1999 [cited 2 May 2002]. Available from Internet: http://www.psc.edu/general/software/packages/genbank/genbank.html.

PORT, Otis. The Next Web. *BusinessWeek*. 4 March [online]. 2002 [cited 25 April 2002]. Available from Internet: http://www.businessweek.com/magazine/content/02_09/b3772108.htm.

SETUBAL, Joao Carlos and MEIDANIS, Joao. *Introduction to computational molecular biology*. Boston; PWS Publisher, 1997. 296 p. ISBN 0534952623.

SIMON, Hank. XML: Data about data. *Modern Drug Discovery* [online].1 March 2001, vol. 4, no. 3 [cited 2 May 2002]. Available from Internet: http://pubs.acs.org/subscribe/journals/mdd/v04/i03/html/03sites.html. ISSN 1532-4486.

SOL, Selena. Introduction to Databases for the Web. *Web Developer's Virtual Library* [online]. 23 November 1998 [cited 2 May 2002]. Available from Internet: http://www.wdvl.com/Authoring/DB/Intro/toc.html.

White, Julia A., MALTAIS, Lois J., and NERBERT Daniel W. An increasingly urgent need for standardised gene nomenclature. *Nature Genetics*, *Nomenclature* [online]. 2001 [cited 2 May 2002]. Available from Internet: http://www.nature.com/ng/web_specials/nomen/nomen_article.html

# APPENDIX

## Figures

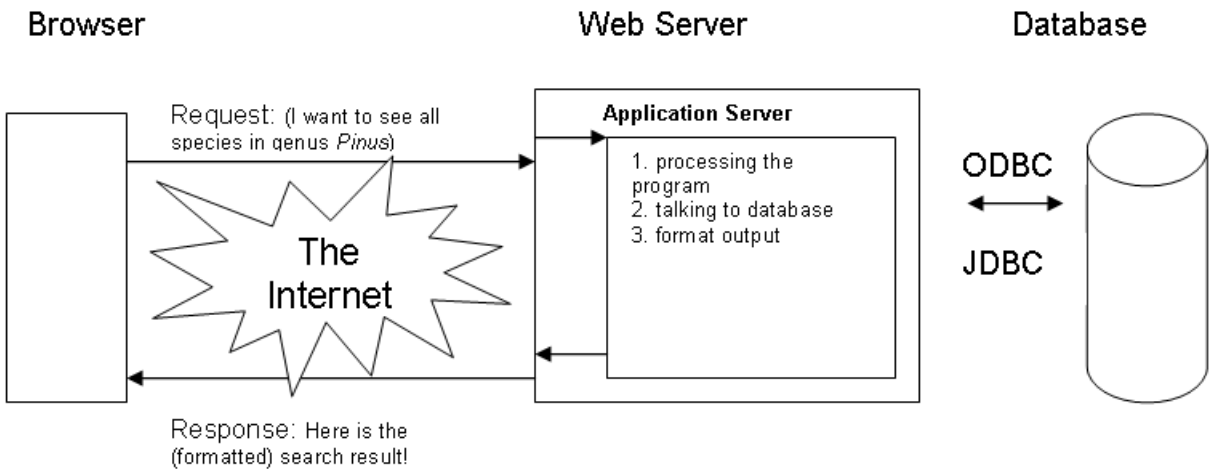


**Figure 1. A process for a client to get information from a database through the Web.**