DATAMAPX: A TOOL FOR CROSS-MAPPING ENTITIES AND ATTRIBUTES BETWEEN BIOINFORMATICS DATABASES

by

Krishna M. Kanchinadam

A Thesis
Submitted to the
Graduate Faculty
of
George Mason University
in Partial Fulfillment of
The Requirements for the Degree
of
Master of Science
Bioinformatics

Committee:

Dr. Jennifer Weller, Dissertation Director

Dr. Don Seto, Committee Member

Dr. Jason Kinser, Committee Member

Dr. Saleet Jafri,
Department Chairperson

Dr. Peter Becker, Associate Dean
for Graduate Programs, College
of Science

Dr. Vikas Chandhoke, Dean,
College of Science

Date: 04/23/2008

Spring Semester 2008
George Mason University
Fairfax, VA

DataMapX: A tool for cross-mapping entities and attributes between bioinformatics databases

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science at George Mason University

By

Krishna M. Kanchinadam
Bachelor's in Engineering (CS&E)
Bangalore University, 1995

Director: Dr. Jennifer Weller, Assistant Professor
College of Science

Spring Semester 2008
George Mason University
Fairfax, VA

ACKNOWLEDGEMENTS

I would like to thank my thesis director, Dr. Jennifer Weller, for her continual guidance throughout my thesis work. She provided valuable input at every step, supervised my work, helped me in the interpretation of the results and assisted me in the preparation of the manuscript. She responded promptly to any questions I had and always encouraged me with positive comments.

I am grateful to Dr. Jason Kinser and Dr. Donald Seto for agreeing to be on the thesis committee and for taking the time to review my work.

Finally, I want to thank my family for their never-ending patience and encouragement, without which it would have been literally impossible to reach this stage.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

DATAMAPX : A TOOL FOR CROSS-MAPPING ENTITIES AND ATTRIBUTES BETWEEN BIOINFORMATICS DATABASES

Krishna M. Kanchinadam, M.S.

George Mason University, 2008

Thesis Director: Dr. Jennifer Weller

Bioinformatics databases are both syntactically and semantically heterogeneous, reflecting, in this inconsistency of their models, the individual interpretations scientists place on the underlying, highly networked, relationships. Data presentation often interferes in the investigators ability to identify elements that partially or wholly map to the same attribute or entity. For example, there are a plethora of databases with public interfaces by which researchers make available subsets of data about genes with characteristics such as the source genome, locus position and allele variant positions, and expression levels, but the names, identifiers, units and chromosomal locations often differ. Since neither the presentation formats nor the nomenclatures are standardized, merging the data can be very complicated, often requiring multiple reformatting steps. At the same time, new experiments often demand a recombination of data from many sources, requiring that the investigator resolve data type and naming inconsistencies, and

often that s/he change relationships as well. While some databases have open source schemas and data, this still leaves a large task for reformatting the data.

Presented here is a tool that facilitates the process of cross-mapping data, when the goal is to populate a second database with a specified subset of information from a source database. The tool is very generic, so we provide use cases both to demonstrate the need and to provide nice tutorials for guiding users through the application. We focus on combining data from arrays that are used to measure either gene expression or genotype information. Each array type has a different interface for reporting the location and composition of probes (sometimes to a subset of community standards, such as the MIAME standard for expression arrays). Gene location, sequence variants and probe locations with respect to those attributes are cross-mapped, giving insight into probes used to assess its gene expression overlap with known SNP genotype information and SNP chip probe information.

The overall goal of this project was to provide a data integration tool by which a researcher can:

1. Access a variety of databases,
2. Provide the correct nomenclature mapping, and
3. Incorporate the information into a common resource that allows data from different experiments and experimental platforms to be correctly combined and then statistical tests applied.

INTRODUCTION

Enormous amount of biological data are shared between various databases. There are a plethora of databases with public interfaces that allow for sharing of complete and/or a subset of the data, as evidenced by the more than 1000 molecular biology databases featured in the annual Nucleic Acids Research database edition each January. Many data formats have emerged, for reasons ranging from facilitating data upload using programmatic tools, to ease of data presentation for database browsers, to data exchange between two distinct databases or data sources. In addition, as the diverse practitioners of bioinformatics apply their idiosyncratic data analysis tools, the resulting output is also delivered in varying formats.

Data access and exchange plays a pivotal role in many types of applications [2]. It generally involves querying a data source and retrieving the results in one of that sources available data format, and then performing a conversion to allow the next task. Also, many important questions in bioinformatics can be addressed only by combining data from multiple databases, since information about a given biological entity is often scattered across different specialized databases. Many types of scientific investigations require combining the data from multiple experiment types in order to answer biological questions effectively and correctly [1]. An example is to combine gene expression

profiles and protein-protein interaction data sets to infer the functionality of a particular gene. This involves extracting information from the two separate data sources – gene expression data, and protein-protein interaction (PPI) data.

Combining data from multiple sources requires careful filtering and an understanding of how the entities and attributes have been defined, because different databases contain overlapping and/or redundant information as well as unique portions. In this situation, effective combination of data from multiple databases also allows for cross-validation and verification of the data, to unambiguously identify duplicated information [1].

In this document, we describe an application that has been developed to provide a mechanism by which a researcher can facilitate the process of data exchange in order to effectively and quickly produce a data warehouse that supports his/her own research. This tool will allow the researcher to access relational databases built on two of the most widely used Open Source database management systems (MySql and/or PostgreSql), execute queries to retrieve the desired result set, provide the correct nomenclature mapping, and then export the resulting data set into a file as comma-separated and/or tab-separated values. This exported result set can then be imported to the local data model, or alternatively the files can be pre-processed in some way, such as applying statistical tests, or filters, prior to storing in a database.

DEFINITIONS

Data Object:

An element that embodies a description of a biological system at a particular level of granularity. For example, a gene, a control region within a gene, an exon, a transcript, a gene product, or a protein-protein interaction of the gene product [1].

Database:

A database is a structured collection of records or data objects stored in a computer system. It relies upon software to organize the storage of data. Thus, the software models the database structure in the database models [8].

Database Model:

A database model is a specification that describes how a database is structured and used [8].

Dataset:

A dataset is a collection of data [9]. It could also refer to a specific version of a database. For example, Swiss-Prot version 39.0 is a dataset for the Swiss-Prot database [1].

Database Management System (DBMS):

A database management system is a software system for storing and querying collection of data.

Relational Database Management System (RDBMS):

A relational database management system is a DBMS that is based on a relational model as introduced by E. F. Codd. Data are stored in the form of tables and relationships among the data are also stored in tables, so the system is self-describing. Most of the popular commercial and/or open source databases in use are based on the relational model [8].

MySql:

MySql is a multi-threaded, multi-user, open-source RDBMS. It provides API's for a variety of languages in addition to OLE DB and ODBC providers/drivers for interaction with the Microsoft .NET environment. It is a popular alternative to proprietary database systems like Oracle and MS Sql Server. In addition, it can run on Unix, Windows and Mac OS, which makes it highly portable from one platform to another.

PostgreSql:

PostgreSql is an open-source RDBMS that runs on all major operating systems. It has, similar to MySql, native programming interfaces for a variety of languages. It is also one of the popular alternatives to proprietary DBMS like Oracle and MS Sql Server.

Microsoft .NET Framework:

The .NET Framework is a software component part of the MS Windows Operating System, providing a base class library that encompasses a large number of pre-coded solutions for common programming requirements. It manages the execution of programs written specifically for the .NET environment.

The .NET Framework consists of two parts: the Common Language Runtime (CLR) and the Framework Class Library (FCL). The CLR is the foundation of the .NET framework. It is used by a variety of programming languages to manage the execution of code, provide memory management, ensuring type safety, and providing a virtual execution system while providing security and code robustness.

The Framework Class Library is a library of classes, interfaces and value types that are included in the .NET framework. This library provides access to system functionality and is designed to be the foundation on which the .NET framework applications, components and controls are built.

The .NET framework allows for a variety of application types to be developed, including Web Applications, Web Services and Windows Forms/client server applications.

.NET Data Providers:

A data provider in the .NET framework serves as a bridge between an application and a data source. A data provider is used to retrieve data from a data source and to reconcile changes to that data back to the data source. DataMapX utilizes the native .NET data providers for MySql and PostgreSql.

The C# Language:

C# (pronounced as C-Sharp) is a type-safe, component-based, high performance language designed by Microsoft for the Microsoft .NET framework. It is derived from C and C++ and closely resembles the Java Programming Language. It is widely accepted as the programming language of choice when developing Microsoft .NET framework applications, irrespective of the application type (Web application, web service and/or a desktop-based client server application).

SharpDevelop – The Free IDE:

SharpDevelop is a free Integrated Development Environment for developing applications in C# on the Microsoft .NET platform. It is open-source and available for download in both source-code and executables from its website **[**9**]**.

IMPLEMENTATION

The design philosophy of DataMapX incorporates the principles of simplicity and accessibility. Usability of the application was given considerable emphasis since this is intended as a tool for non-database experts, specifically scientists who have a great need to produce a data warehouse to support a particular project and little interest in becoming database management experts. This section describes the design and coding framework for the application.

Overall Requirements:

The application must allow access to a variety of databases. We have targeted MySql and PostgreSql as the two target database management systems to be implemented in the application. One of the primary reasons is the ready availability of both databases. Both are full-featured and popular alternatives to proprietary databases. These databases are those most commonly used in the bioinformatics community, because they allow users to employ robust databases at minimal or no cost.

The application should provide the users the ability to connect to any supported database, by providing the appropriate credentials to login to the database. The application should

allow the user to select a specific database and view the tables and views present in the database. This allows the user to select the appropriate tables and attributes, frame a SQL statement appropriately, execute it against the selected database, and then view the resultant records as a quality check, and finally allow export.

It is quite common in the bioinformatics community to query for multiple thousands of records, with large amounts of data per record, so the application should be able to handle large volumes of data.

The application should provide a mechanism to export a retrieved result set into an appropriate format that will allow the researcher to export it into their local database schema with minimal problems, and/or apply tests as appropriate. We have chosen to export the data into a comma-separated and/or tab-separated file (CSV and/or TSV). The generated file can be opened in any text editor and/or Microsoft Excel, size permitting.

Schema Requirements:

It was determined that the application should not have its own database schema, in keeping with the principle of simplicity. The application is intended purely as a tool that will connect to existing databases, to improve interoperability while keeping the investigator completely in control. This will encourage adoption of the tool, as the researcher does not have to worry about the complexity of managing a separate database used by the application in addition to those of primary interest.

Architecture:

DataMapX is written in C# programming language, and implemented as a standalone .NET executable targeting the Microsoft .NET Framework version 2.0 or above. Implementing DataMapX as a standalone desktop based application avoids the problem of session time-outs inherent when using web-based application and performing long-duration operations.

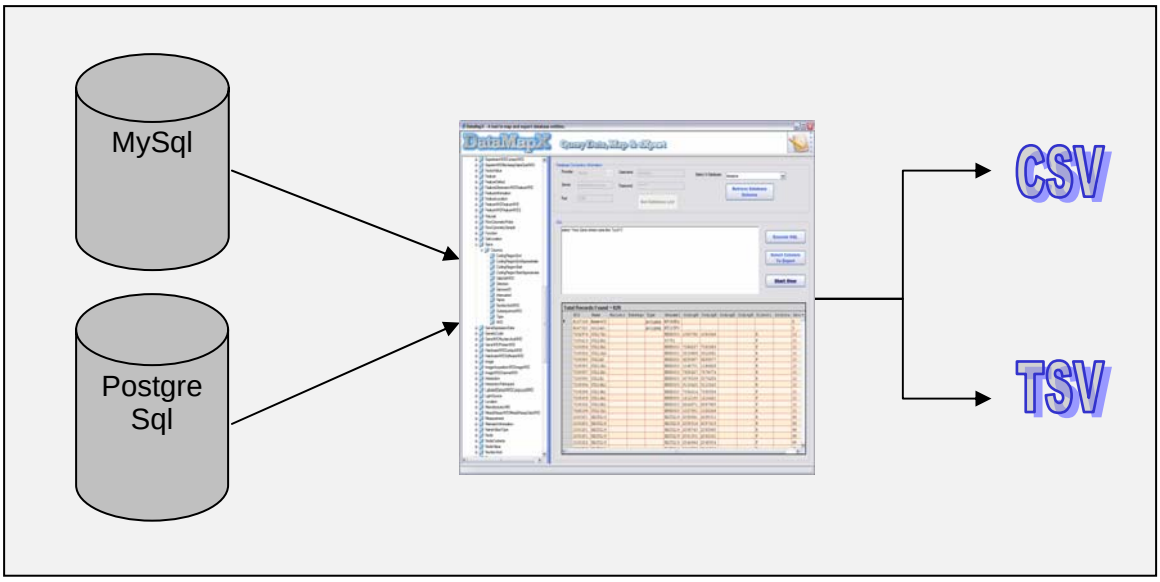

Figure 1: Basic Architecture of DataMapX

The figure above shows the basic architecture of DataMapX. The application queries any MySql and/or PostgreSql database and generates an output file containing the dataset a researcher wants to export, in an appropriate output format.

Workflow:

Figure 2: DataMapX Workflow displays the workflow for DataMapX. A researcher will connect to the appropriate MySql and/or PostgreSql DBMS and choose to view the tables and views in a database. They can then run SQL queries against the chosen DBMS and view the results in a tabular data grid. They can perform a quality check of the retrieved results before exporting the results or a subset of the results to a file in the desired output format.

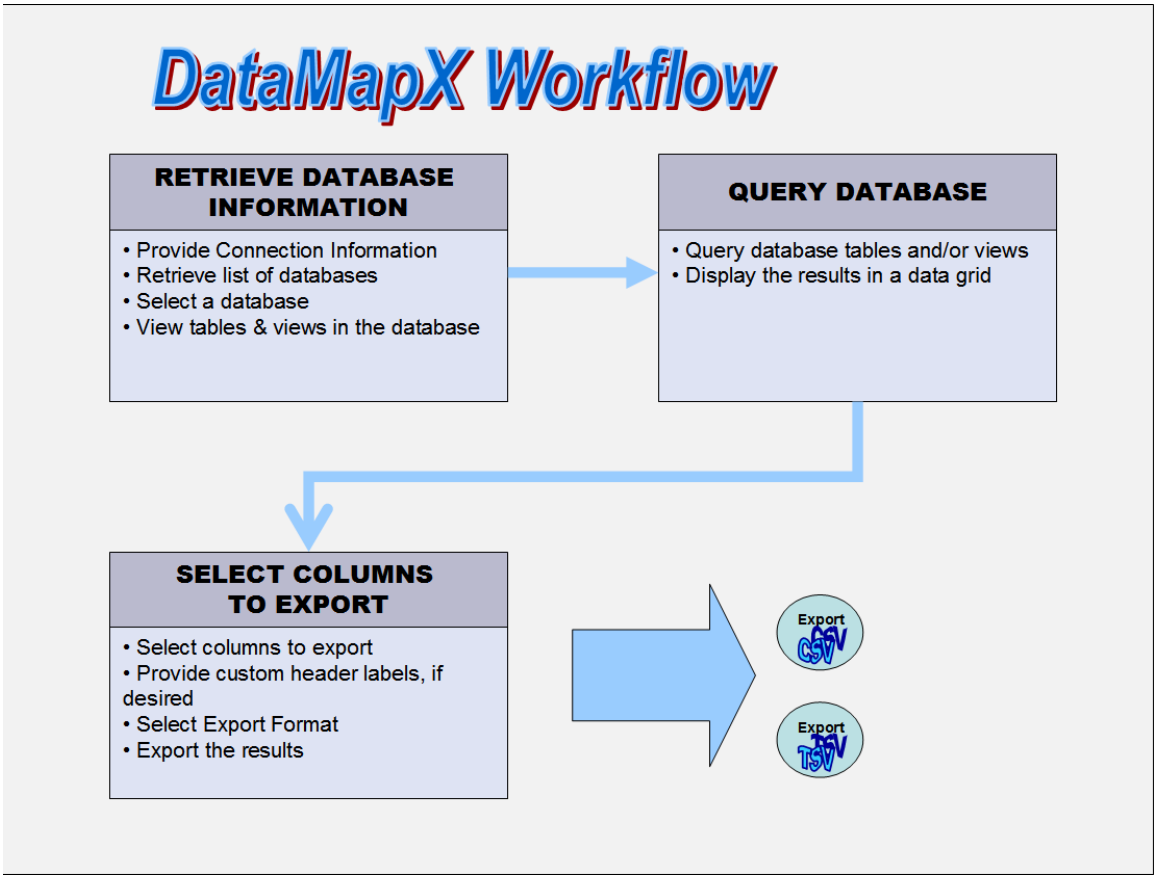

Figure 2: DataMapX Workflow

User Interface:

DataMapX is packaged with an installation program to facilitate the process of installing the application on a client computer system. After successful installation, the application can be executed via the Windows Start Menu → All Programs option. On executing DataMapX, the main window of DataMapX is displayed (See **Figure 3:** DataMapX Main Window below).

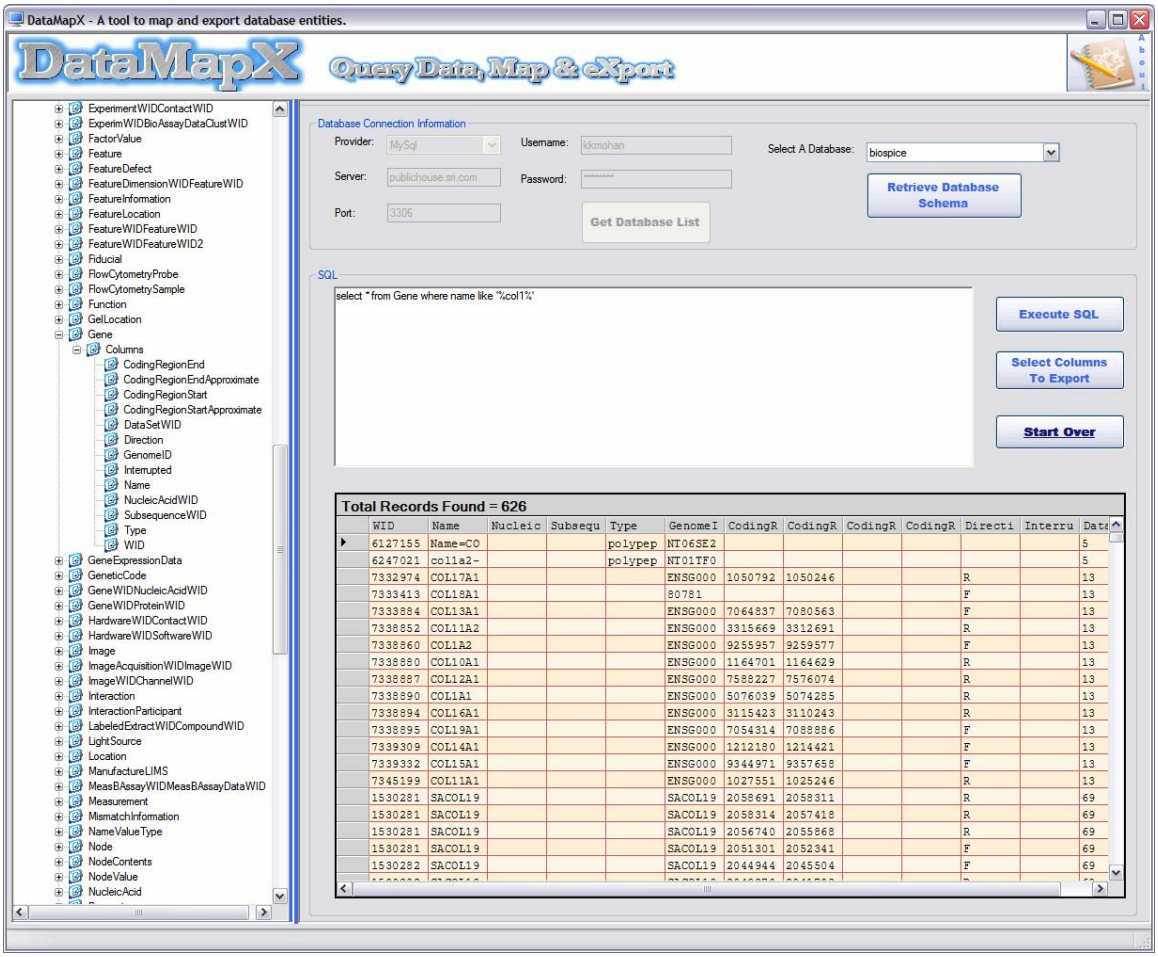

Figure 3: DataMapX Main Window

The user of the application will select a DBMS and provide information to connect to it. The application retrieves a list of databases in the DBMS and provides the user an option to select the desired database. The database schema can be retrieved and displayed in a hierarchal manner, thus allowing the user to view the database schema visually, before attempting to frame a SQL statement and execute it against the selected database. Executing a SQL statement retrieves the result and displays it in a tabular format in the data grid, thus allowing the user to browse through the results before attempting to export the desired information into a supported file format.

Figure 3: DataMapX Main Window above displays the main screen of DataMapX connected to PublicHouse, which is a publicly queryable warehouse of biological databases available over the internet. PublicHouse is built on top of MySql, and provides a mechanism for large-scale data mining using SQL statements over the Internet. In the example above, once connected to the PublicHouse database, a user of DataMapX queries the gene table, and the results are displayed in the datagrid.

Once the researcher is satisfied with the retrieved results, s/he can choose to export the dataset. Figure 4: Select Columns to Export displays the window that allows a user of DataMapX to export either all or a subset of the dataset columns. The user has the ability to:

1. Select the columns to be exported, and

2. Modify the column header text in the output file, to conform to the names in the target database.

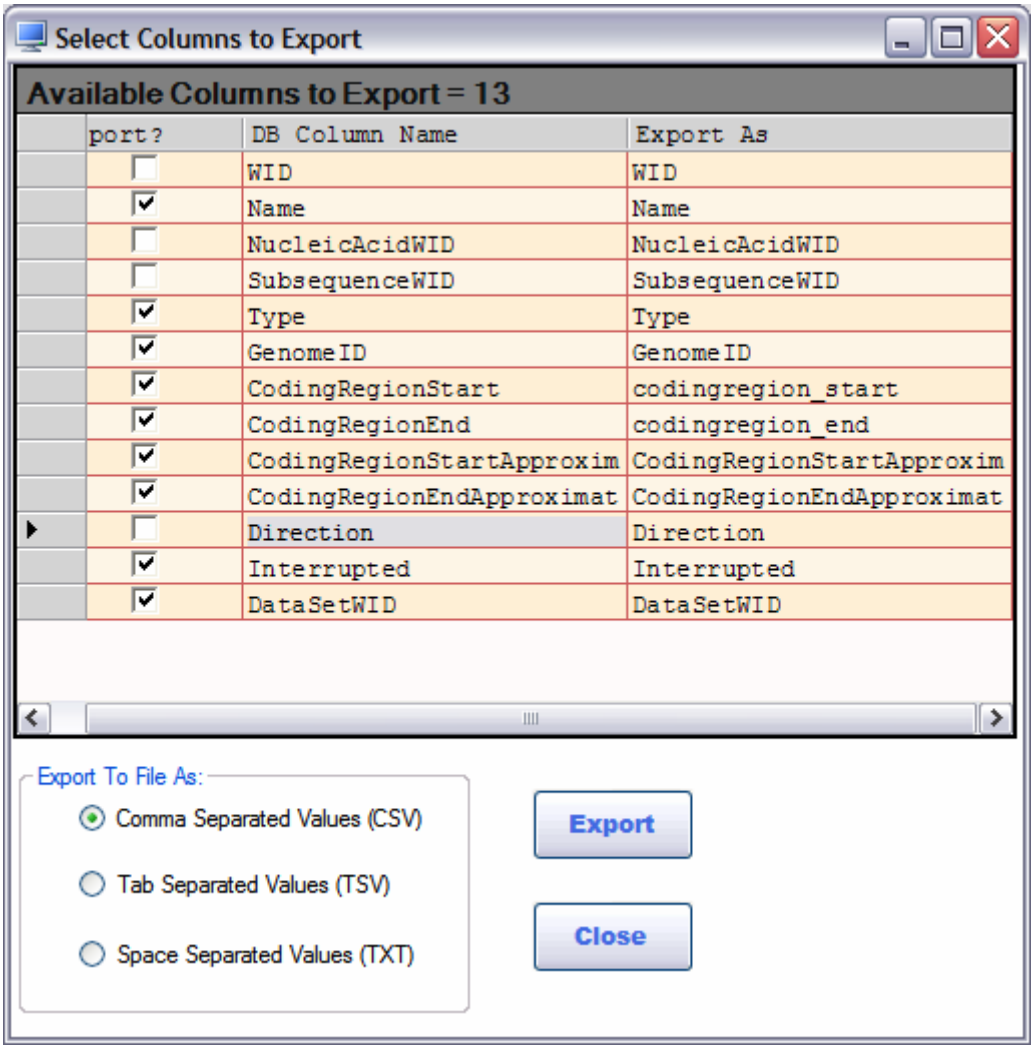

Figure 4: Select Columns to Export

The exported dataset can be saved in text format on the local system, opened and viewed in any text editor and/or Microsoft Excel.

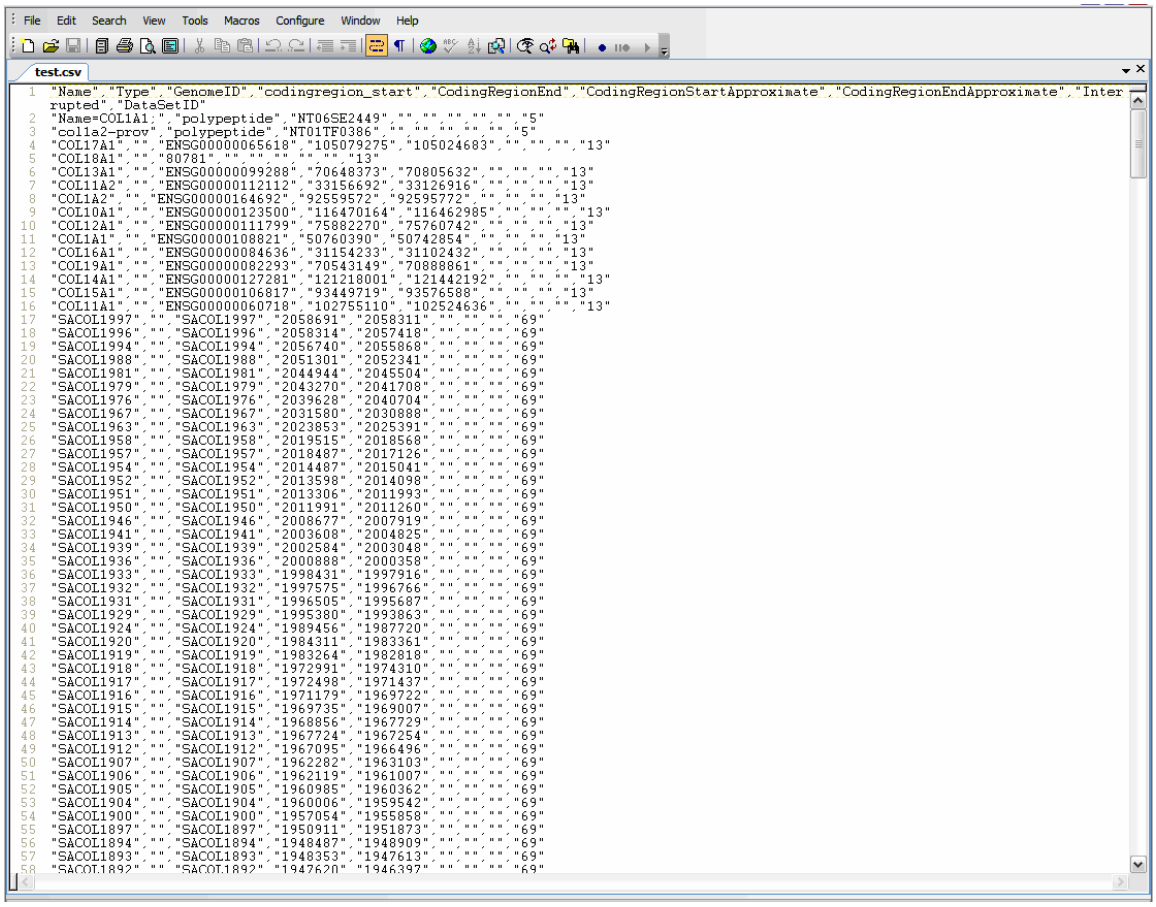

Figure 5: Exported Dataset

Note: The exported file can be opened in Microsoft Excel. However, Excel has a size restriction that prohibits it from displaying more than 65536 records in a single worksheet. Thus, if the exported dataset exceeds 65, 536 records, Excel will only display a subset of the exported dataset.

System Specifications:

The following tools, utilities and technologies were used in developing DataMapX:

1. MySql RDBMS 5.x.

2. PostgreSql RDBMS 8.1.x.
3. C# Programming Language, and the .NET Framework 2.x.
4. SharpDevelop IDE 2.x – An open-source, freely available IDE for developing .NET framework applications.
5. NSIS (Nullsoft Scriptable Install System) – A professional open source system to create Microsoft Windows installer programs.
6. HM NIS Edit – A free IDE/editor for NSIS.

TESTING AND RESULTS

The use cases are driven by a real research need: a group at the Carolina Medical Center in Charlotte, NC, is interested in doing a study on the development of hernias during the process of wound healing in surgical patients. Extensive clinical evidence shows no correlation of the development with a variety of factors such as age, gender, type of surgery, secondary infections, and place of surgery or surgeon. Thus the clinicians would like to perform a genome-wide association study, looking at single nucleotide polymorphic markers. Through extensive literature searches, a set of 35 genes with functions highly relevant to the processes of tissue damage/healing have been drawn up, as initial targets. The genes and their numbers are shown in the table below (Table 1: Candidate Genes Involved in Wound Healing).

The original goal of the project was to use RT-PCR assays for each of the SNPs. Further investigation indicates the number of SNPs uncovered by the HapMap investigators has more than doubled the set of associated SNPs for these genes in the past two years. There are a number of high-throughput platforms that assay SNPs in parallel, such as the Affymetrix SNP6.0 arrays. There are also arrays that assay the expression of the exons of each gene, some of which contain SNPs not in the genotyping arrays. One of the questions the investigators would like to have answered is how many SNPs each of the

various platforms measure for each of the genes of interest. A second question that arises is whether the target gene list is sufficient: the genes selected may be in pathways that are important but may not be themselves the points of variation. While some of the array platforms allow all genes to be queried initially, they are very expensive, so it would be useful to discover how many genes are in the principal pathways to which set of genes belongs.

Table 1: Candidate Genes Involved in Wound Healing

Candidate Genes Involved in Wound Healing

	Gene Symbol	Gene Name	Size (bp)	# of SNPs
Collagen				
	COL1A1	Collagen IA	17,538	103
	COL1A2	Collagen IA	36,673	162
	COL3A1	Collagen III	38,375	127
	MMP1	Matrix Metalloproteinase 1	8,245	98
	MMP9	Matrix Metalloproteinase 9	7,654	60
	MMP10	Matrix Metalloproteinase 10	10,112	34
	MMP11	Matrix Metalloproteinase 11	11,468	31
	MMP13	Matrix Metalloproteinase 13	12,740	31
	TIMP1	Tissue Inhibitors of MMP 1	4,358	6
	TIMP2	Tissue Inhibitors of MMP 2	72,413	270
	TIMP3	Tissue Inhibitors of MMP 3	62,221	2,314
	TIMP4	Tissue Inhibitors of MMP 4	6,079	15
	BMP1	Bone Morphogenic Protein 1 3	6,093	71
	LOX	Lysyl Oxidase	11,937	45
Inflammation				
	IL-1a	Interleukin-1 alpha	11,480	69
	IL-1b	Interleukin-1 beta	7,021	62
	IL-6	Interleukin-6	4,798	47
	IL-8	Interleukin-8	3,158	23
	TNFa	Tumor necrosis factor alpha	2,764	28
	SOCS1	Suppressor of cytokine signaling 1	1,767	6
	IFNg	Interferon gamma	4,973	2
Growth Factors				
	TGFB	Transforming Growth Factor beta	23,167	102
	EGF	Epidermal Growth Factor	99,371	542
	PDGF	Platelet Derived Growth Factor	21,273	98
	VEGF	Vascular Endothelial Growth Factor	14,392	85
	FBGF2	Fibroblast Growth Factor	71,515	250
Coagulation				
	uPA (PLAU)	Urokinase Plasminogen Activator	6,366	37
	tPA (PLAT)	Tissue Plasminogen Activator	32,441	280
	uPAR	Urokinase Plasminogen Activator Receptor	21,772	179
	BDK	Bradykinin	26,623	267
	ACE	Angiotensin I Converting Enzyme	20,547	167
Transcription factors				
	STAT3	Signal Transducer & Activator of Trans 3	75,172	320
	NFKB1	Nuclear Factor Kappa B (p105)		414
	NFKB2	Nuclear Factor Kappa B (p100)		26
	RelA	Nuclear Factor Kappa B (RelA)		36
	RelB	Nuclear Factor Kappa B (RelB)		69

Use Case 1:

For each gene in the list, first acquire the gene model information, including the chromosomal location, strand on which the gene is code, and the positions of the exons, introns and untranslated regions (UTRs). Either the Entrez or Ensembl databases serve this information, although the frames of reference differ and the format in which the data is made available are different. Subsequent to this, we find the most recent information for the positions and alleles of the known SNPs in the coding region (from NCBI's dbSNP database). Finally, for each of the array platforms of interest, determine the location of each of the probes for each array type, for each of the genes, in this case the Affymetrix SNP6.0 array, and the Affymetrix Exon Array (this set of information can be acquired from the ENSEMBL, UCSC and NetAffx sites, but in most of them the information is graphical). This will allow the investigator to determine how many of the known SNPs are measured by each platform, and how many would be missed. This use case supports an experimental design function.

Use Case 2:

For the list of genes, find with what other genes they share pathways. There are a number of pathway databases, including KEGG and HumaCyc, the human-centric version of MetaCyc. MetaCyc databases are freely available to academics, from SRS, in a relational DBMS. Given the enlarged set of genes for the entire pathway, an additional step, in order to be sure that the genes are functionally relevant, is to query for known mutations

in the Online Mendelian Inheritance in Man (OMIM), and, if any appear to be relevant to healing, they can be added to the set, after which one repeats the steps outlined in Use Case 1.

Availability of Code and Data:

The database schema, the input data sets (as files, with the SQL statement used for retrieval when the source allowed such) and the application code are provided on a Supplementary Materials CD.

DISCUSSION

While computational biologists focus primarily on algorithm development, bioinformaticians spend a great deal of time acquiring data sets relevant to the current hypothesis, cleansing that data of errors, mistakes and replicates, and often performing statistical tests to determine whether assumptions about sample and population structures are valid before proceeding to the 'real' research. In the wet lab the majority of effort goes into preparing reagents and performing controls, and the data preparation steps have a similar role and time requirement in the bioinformatics experiment. Many bioinformaticians have survived on flat-files and directory structures for managing related datasets, in part because learning the schema and definitions of all the contributing databases, or the skills needed to maintain them independently and make them interoperable can be daunting. This is especially true when the source databases are extremely complicated, like the MIAME-compliant databases (180-200 tables) or dbSNP (100 tables). If one only wants six of the attributes, the cost-benefit analysis can seem heavily weighted to extracting the bit wanted and simply dealing with flatfiles for the data manipulation and analysis steps.

On the other hand, the tools for developing a local data warehouse have become progressively more user-friendly, so the time investment in learning to set up a personal

database is no longer months, but weeks. The benefits of being able to perform set queries to recombine data in new ways, add new dimensions quickly, and use built-in functions for low-level data filtering are nearly as attractive as the notion of managing dozens of tables having simple names as opposed to thousands, or tens of thousands, of related files whose names by necessity are obscure (numbered) or complicated (concatenated related data set names). Since many investigators are making their data available as relational database dumps, along with the schemas, it would seem that this is a research tool that is finally maturing.

Thus, it is timely to provide a tool that allows easy extraction of a desired subset of data from such a source. Simple table dumps (most often the only functionality provided if you don't want the whole thing) require a lot of cutting and pasting, if the target table is not exactly like the source table. This tool allows the investigator to simplify the process and acquire only those elements of interest, using the sophistication of SQL. The other major problem an investigator faces is that data elements almost never share the same name between two databases, unless a true ontology is used, and very few of these exist. This reflects real differences in interpretation, and assumptions, about the data elements, but means that a great deal of re-naming must be done in the recombination process. It is best to do this while the data is still in its original context in the source database, so the tool allows the investigator to provide a new name upon export.

The use cases provided here were for collaborators in the early phase of starting a very large project, so the experimental outcome cannot be discussed at this time. These collaborators, while sophisticated in molecular genetics and biology, do not have the time, or interest, to drill down through the presentation layers of the six source databases used to acquire the data assembled in the project's highly focused data warehouse. This small local warehouse (with four tables), however, is something with which they are very comfortable. In terms of support for the experimental design process, the choice has been to abandon the individual SNP detection approach and use the Affymetrix SNP6 array platform. Because the number of SNPs had grown so fast, the cost-benefit analysis they performed compared the cost of reactions just for these genes versus a whole genome array. The pathway analysis, while considered interesting, became irrelevant, since all of the genes are represented on the SNP6 array. However, if there had been fewer SNPs in the target genes and if some of the genes in the pathways were shown to have relevant mutations, then this would have convinced the research team to add those genes, according to them.

The application will be used by Dr. Weller in her Bioinformatics Databases course in future years, as it provides a useful tool in populating a data warehouse project, required of the students, that is currently missing. Students spend a great deal of time manipulating files in order to prepare data for upload to their project databases. One of the values of the tool for the course is the visualization of the process, from the source database to the

extracted attributes, and the guidance in attribute selection and renaming, which encourages students to think ahead to where they want to use the data.

A final piece of functionality that will be added to the tool before public release is an XML conversion function. More and more of the ‘omics’ data sets have an associated XML format, for data exchange purposes. The appropriate XML Writer will supply the tags deemed appropriate by the user, allowing the data to be mapped to another system by the linked XML Reader. While we cannot supply all possible versions (there are three for MAGE-ML alone) a demonstration of one of them will allow other developers to provide the functionality for the groups with whom they work.

This project started out having a very narrow focus: to map Affymetrix probes on the U95 and U133 arrays to loci in dbSNP. As the development proceeded the realization emerged that with small changes (less hard-coding and more guidance of the user) a far more flexible tool could be developed.

CONCLUSION

DataMapX is an application that provides an easy, visually guided means of cross-mapping entities and attributes between multiple bioinformatics databases. The software application facilitates the process of cross-mapping and combining data from multiple databases, running quality checks on the retrieved results, and finally exports the results, or a subset of the results, into a second database.

The software is available as a standalone .NET executable and can be downloaded and installed by users who want to utilize the application for their own laboratory-specific needs.

Availability and Requirements:

- **Project Name**: DataMapX
- **Project Home Page**: http://datamapx.kanchinadam.com
- **Operating System(s)**: Microsoft Windows
- **Programming Language**: C#
- **Other requirements**: Microsoft Windows .NET Framework 2.x or above

APPENDIX

The source code, input and output datasets are available in the accompanying CD media, in addition to being made available on the project web site.

REFERENCES

REFERENCES

1. BioWarehouse: a bioinformatics database warehouse toolkit. Thomas J Lee, Yannick Pouliot, Valerie Wagner, Priyanka Gupta, David WJ Stringer-Calvert, Jessica D Tenenbaum and Peter D Karp.

2. Data Integration in Bioinformatics Using OGSA-DAI. Shirley Crompton, Brian Matthews, Alex Gray, Andrew Jones, Richard White, available online at http://www.allhands.org.uk/2005/proceedings/papers/500.pdf.

3. Susan B. Davidson. OMICS: A Journal of Integrative Biology. January 1, 2003, 7(1): 11-12. doi:10.1089/153623103322006490, available online at http://www.liebertonline.com/doi/abs/10.1089/153623103322006490.

4. Nucleic Acids Research database Journal: http://www3.oup.co.uk/nar/database/c/

5. NCBI's Single Nucleotide Polymorphism database (dbSNP), available online at http://www.ncbi.nlm.nih.gov/projects/SNP/.

6. International HapMap project, available at http://hapmap.org.

7. Achard, F., et al. (2001) XML, bioinformatics and data integration. *Bioinformatics*, **17**, 115–125, available online at http://bioinformatics.oxfordjournals.org/cgi/content/abstract/17/2/115.

8. Wikipedia.org, a web-based, free content encyclopedia project.

9. #develop (short for SharpDevelop), an open-source free IDE for developing applications on the Microsoft .NET platform. Project Web Site: http://www.icsharpcode.net/OpenSource/SD/Default.aspx.

CURRICULUM VITAE

Krishna M. Kanchinadam is a Software Engineer with over 12 years of experience in the Information Technology field. He holds a Bachelor's degree in Computer Science & Engineering from India.