Improving Functional Analysis of Microarray Gene Expression Data

by

Michelle Bobo

A Master's Thesis

Presented to the Division of Medical Informatics

and Outcomes Research

Oregon Health & Science Univeristy

In partial fulfillment of

the requirements for the degree of

Master of Science

August 2002

School of Medicine

Oregon Health & Science University

CERTIFICATE OF APPROVAL

This is to certify that the

Master's thesis of

Michelle Bobo

has been approved

[redacted]

Committee chair

[redacted]

member

[redacted]

member

Table of Contents

Abstract..iii

1. **Introduction**..1

2. Background & Significance
2.1 What is microarray research and why do it? 2
2.2 Microarray workflow and information needs..3
2.3 Gene Annotations..8
2.4 Gene Ontology..10
2.5 Microarray Sequence Annotation Resources..12
2.6 Project Setting..13
2.7 Specific Project Aims..14

3. Methods
3.1 Data Warehouse..17
3.2 Annotation Delivery..20
3.3 Pathway Modeling..25
3.4 Methods Summary..29

4. Results
4.1 Data Warehouse..30
4.2 Annotation Delivery..33
4.3 Pathway Modeling..36

5. Evaluation & Discussion
5.1 Evaluation of Project Components..38
5.2 Current Developments in Bioinformatics..43
5.3 Project Conclusion..45

Appendix I – Warehouse Dataset Loading Protocols
A. PDIP Warehouse Schema..47
B. NCBI UniGene and LocusLink 48
C. Jackson MGI 50
D. Riken FANTOM..51

Appendix II - Protocols for using annotations
A. Chip_annotations View in Data Warehouse..53
B. Loading annotations into OmniViz..54
C. GenMAPP 56

Appendix III - QChip
A. QChip HTML Query Form..58
B. QChip Servlet Source Code..59
C. SpreadsheetFormat Servlet Source Code..64
D. Additional Documentation..68

References..69

Acknowledgements

I would like to thank the members of my committee: Christopher Dubay, Srinivassa Nagalla, and William Hersh for guidance through this thesis process. I would also like to thank Mark Turner for many hours of advice and instruction.

Finally, a big thank-you to my family, friends, and fellow students for support along the way.

Abstract

Microarray gene expression research frequently generates large numbers of questions or information needs throughout the course of the experiement. Many of these questions relate to the nature of the sequences on the microarray. In order to effectively analyze experimental data, it is necessary to maintain significant information about each of the 12,000 or more gene transcripts represented in the experiment.

This thesis describes the development of a knowledge resource of DNA sequence functional annotations, intended to aid researchers conducting microarray experiments. The resource includes an integrated data warehouse of sequence annotations for human and mice, and data access tools for retrieving and utilizing the available information.

The data warehouse maintains up-to-date sequence annotation data from multiple authorities. The core of the warehouse is the NCBI UniGene and LocusLink sequence descriptions. Additionally it contains data sets of Gene Ontology annotations of mouse sequences from both the Jackson Laboratories and the Riken organization in Japan.

In order to optimize the effectiveness of the warehouse to researchers with diverse needs, it was necessary to provide several means of accessing the information. These include spreadsheet files, scripts to import data into analysis programs, and a web-based query interface. Additionally there is documentation which demonstates and discusses tools designed to describe microarray data as a picture of meaningful biological event.

The tools that have been created are designed to be easily modified to accommodate changing needs in the research environment. Additionally, protocols and documentation are provided to enable maintenance and updates. The combined resources will improve functional analysis of microarray data and represent a significant addition to the informatics tools available to the Department of Pediatrics Biotechnology Core Facility.

Research Question: Can we use bioinformatics techniques, including pathway visualization, to improve researchers' ability to analyze microarray gene expression data according to gene function?

1. Introduction

The purpose of this project is to develop a set of bioinformatics tools and protocols which will be used by biomedical researchers to facilitate analysis of microarray gene expression data according to gene function. The components of the project include development of an integrated data warehouse of gene functional annotations, tools to access the sequence annotations including a web-based information retrieval system, and a demonstration of a new analysis technique providing visualization of biological pathways. The resulting tools and protocols will be used and evaluated by researchers conducting microarray gene expression analysis experiments in the Department of Pediatrics Center for Biomarker Discovery.

2. Background & Significance

2.1 What is microarray research, and why do it?

Microarray gene expression studies are an important component of the disciplines of both genomics and systems biology, which are widely considered to be on the cutting edge of biomedical research [1]. Genomics is the study of an organism's entire genome, and microarray research uses genomics to investigate the dynamic and temporal patterns of gene expression in that species. Systems biology has been defined as the quantitative study of biological processes as integrated systems instead of isolated parts [2]. The goal of both fields is to build upon the recent achievements of sequencing the human, mouse, and other genomes and begin to leverage that data into new knowledge for curing human disease.

Microarray gene expression analysis is an important investigative technique because it allows a researcher to measure the expression level of thousands of genes from a single tissue sample simultaneously. By comparing gene expression profiles from sets of samples representing control and experimental conditions, it is possible to determine which genes are expressed differentially, and thus may be related to an experimental condition of interest.

The microarray technique was developed in 1995 by Dari Shelton as his PhD thesis at Stanford University and its' potential was recognized immediately by researchers and biotechnology applications developers [3]. It is currently a very popular and widely used experimental technique.

The microarray itself is usually a glass slide onto which 10,000 – 12,000 discrete "spots" of cDNA are precisely applied by a robot. The slide is then hybridized with fluorescently labelled DNA that has been generated by reverse transcription and amplification of mRNA from a tissue sample of interest. During the hybridization, the labelled DNA from the tissue sample binds to homologous sequences on the microarray slide. The slide is then scanned by a laser and the intensity of fluorescence is read with a confocal microscope and a CCD camera.

Despite its' enourmous popularity, microarray research is by no means simple and still rarely provides conclusive answers. There is not yet a consensus or "gold standard" on exactly how data should be analyzed and evaluated. The nature of the complexity of microarray experiments is that they continually generate a large volume of questions and information needs for the scientists involved. The following paragraphs will briefly describe the experimental process [4] and discuss the information needs likely to arise at each stage.

2.2 Microarray Workflow and Information Needs

The complexity of microarray research inevitably creates a large volume of questions and information needs for the researcher. These questions arise at many stages of the experiment and must be answered in order to effectively analyze the experimental data. [Figure 1, page 7]

Information needs typically begin in the planning and design stages. At this point, the investigator will likely want to know something about the cDNA 'spots' on the microarray.

- What types of sequences are on the slide?
- How many represent well-known genes?
- How many are sequences which are known to be expressed, but not yet characterized as genes (EST's) ?

The investigator will also likely have a list of specific genes of interest and will want to know which of those genes are represented by sequences on the slide.

The actual "bench work" is then the second stage of the experiment. It consists of several steps that are all done according to local protocols [5]. The tissue samples are identified and gathered. They are prepared for gene expression analysis by extracting the mRNA, reverse transcribing it to cDNA and incorporating the fluorescent label or dye. Next the labelled cDNA from the tissue sample is hybridized with the microarray slides. The slide is read by a laser scanner, which generates an image file, and gridding software is used to specify the location of the spots in the image file that are to be quantified. This stage typically does not generate many scientific questions, and could be considered the easiest part of the experiment.

The third stage consists of the early stages of data analysis. From this point on, all of the analysis is performed on the resulting data set using specialized software. After the spots are quantified, a number of mathematical techniques need to be applied to normalize the data, and generate descriptive statistics of the distribution and general quality of the data. It is important to examine levels of signal intensity across regions of the slide, and compare over-all intensity levels with other slides from the same experiment. At this stage, the dominant question for the researcher is going to be, "Is this useable data?"

Stage four is comprised of the major statistical analyses. The researcher (or the hired statistician) determines which genes are significantly differentially expressed across the set of biological samples. A complementary technique uses clustering algorithms to group all of the sequences into clusters representing similar patterns of expression across samples. As this analysis is carried out, the major questions begin. The researcher will likely end up with many sequences, at least 50 or more, which are considered significant in the experimental context, and will then want to know as much as possible about each sequence. Questions about the significant sequences include whether it encodes a known gene, what gene it encodes, and what the function is and what else it interacts with in the cellular environment. If the significant sequence is an EST, the investigator will want to know as much as possible about similarities to known genes in other species, or motifs in the sequence that encode a known protein domain. An equally important question at this point is whether the results really represent a biologically meaningful phenomena.

Stage five is the completion of the data analysis. At this point the researcher needs to interpret and describe the data in a scientifically meaningful context. The aim here is to move beyond a result list of up-regulated and down-regulated genes and produce an interpretation of the results as a biological process. This should inevitably result in the generation of new research hypotheses and the design of additional experiments to test them. Another important activity at this stage is the publication of the experiment results in both public databases and journals. Scientific publishing requires techniques for data "visualization" which translate complex quantitative data into a more intuitive graphical and visual format [6].

The final stage consists of the activities following the completion of the microarray experiment. The researcher will continue to design and conduct additional experiments while also staying current with the scientific literature of their field. It is likely that they will continue to generate scientific questions that can be answered with their data. Or they will want to examine data that other researchers that have published in a public database. Thus the microarray experiment will continue to generate information needs long after it has been completed.

The preceeding paragraphs demonstrate that microarray gene expression research inevitably generates a large volume of questions and information needs for the scientists involved. The variety of types of questions makes it unlikely that any single resource will be able to address or answer all of them, however, there is a great deal that can be done to aid in the process. Many of the big questions essentially boil down to what is known scientifically about the sequences on the microarray chip, which are the focus of the experiment. A major component of this thesis project, therefore, is to develop a local information resource of gene chip sequence annotations that can be used to answer questions arising from the experiment.

Workflow Stage	Questions & Information Needs
(1)Pre-Experiment Planning & Experimental Design	What types of sequences are on this chip? How much is known about the sequences? What % are well known/characterized and what % are EST's? What about these specific genes- are they on the chip? How are we going to analyze the data?
(2) Experiment Sample preparation Hybridization Lazer scan chip Gridding	(this is the easy part)
(3) Pre-Analysis Spot quantification Data quality inspection Normalization Slide distribution statistics	Is this useable data? Is this good data?
(4) Data Analysis Determine statistically significant differentially expressed genes Use clustering algorithms to group data according to patterns of expression	Are these biologically meaningful results? What is known about these significant genes? - sequence, protein product, function and structure, biochemical interactions
(5) Scientific Interpretation Fit a model to the experimental results Generate new biological hypotheses Design and conduct additional experiments Publish microarray experiment results	Why are these genes significant? What else has been published about this sequence? What additional experiments should be done? How can we best display and communicate this result?
(6) Post-Experiment Continue research On-going queries of experimental results Keep up with literature Design new experiments	Did I have data on this gene? What did this gene do in my experiment?

Figure 1: **Stages of Microarray Experiment Workflow and Corresponding Information Needs**

2.3 Gene Annotations

Gene annotations are sets of data representing the information that is known about a particular gene, creating the link between sequence and biology. Historically, annotations were first used to describe meta-data by researchers submitting expressed DNA sequences into repositories such as GenBank. At that time they typically included information about how the sequence was obtained, and possible errors within it. As sequencing techniques evolved and the quanitity of available sequence data grew exponentially, the content and volume of additional annotation information grew rapidly as well. Now that the human genome has been completely sequenced [7, 8] and the mouse genome will be completed soon, the next major task is to annotate them entirely. Consequently, annotation data for human and mouse is currently being produced at a high rate [9, 10].

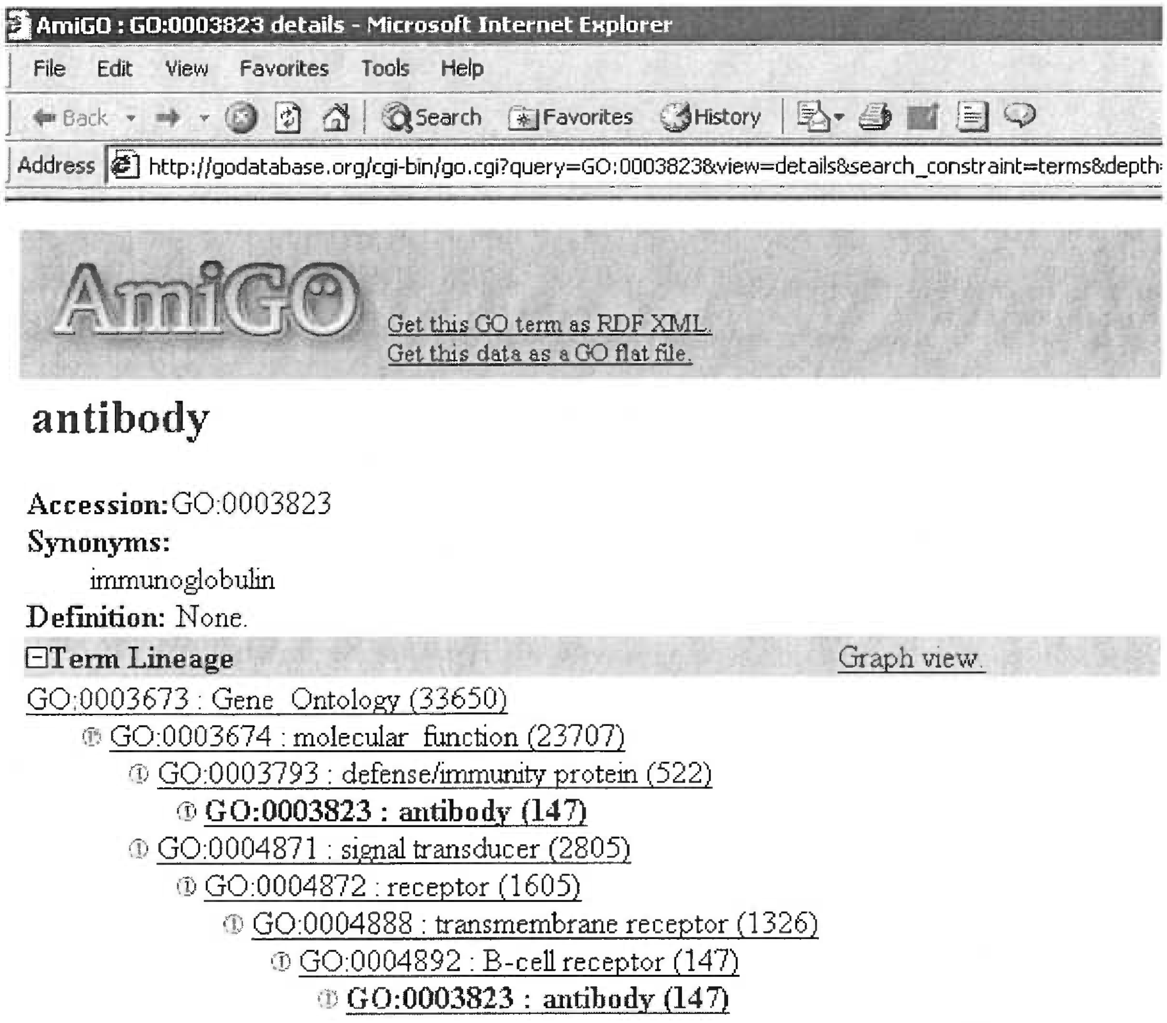
Gene annotations include information about the gene at several levels of complexity. At the simplest level, they include the sequence of the gene, and its' chromosomal location in a particular organism. The next level of complexity is the protein sequence that the gene encodes. It is necessary to determine which genes and proteins in other species are homologous, or similar in sequence and function. From known homologies, and previously published molecular biology data, one can then begin to make inferences about the molecular function of the gene product in the organism; these are called functional annotations. At the process level, annotations describe the major biochemical pathways that the target gene is a part of, and characterize its' interactions with other components of that pathway. This is where the true biological importance of a gene is described [11-13].

A number of publicly available resources exist for accessing gene annotation information. Several major national and international research organizations have taken on the task of researching, compiling, and producing web-accessible databases which aim to ultimately cover the entire genomes of the major research organisms. The most well known is the NCBI (National Center for Biotechnology Information), which is a component of the U.S. National Institute of Health. The NCBI has developed an intricate system of interconnected databases that provide access to an enormous volume of sequence annotation resources. The two NCBI databases that are most important for microarray data analysis are UniGene and LocusLink. UniGene organizes full-length mRNA sequences and EST's into clusters that each represent a single known or putative gene. This is important because a single gene can currently be represented by dozens of GenBank sequence accession numbers, only one of which is likely to be used to identify a specific sequence on the microarray. The LocusLink database provides additional information on the accepted and alternate symbols and names for a particular locus, as well as some functional annotation information. As its' name implies, it also serves as a major link or gateway to other resources for a gene of interest, including homologies to other organisms, diseases known to be related to the gene, and relevant documents in the primary literature. Other important sources of annotation data are the Mouse Genome Informatics group, through the Jackson Laboratories, and the FANTOM (Functional Annotation of Mouse) database through the Japanese group, Riken [14]. The NCBI LocusLink, the Jackson Laboraties, and the Riken group all provide functional annotations using the Gene Ontology vocabulary.

2.4 Gene Ontology

Functional annotations describe the molecular function of the gene product, and its broader role in the cell or tissue type. In order for annotations to be consistent, meaningful, and sharable, it is important to use a standard vocabulary. Fortunately for biological researchers, there exists a single vocabulary that has achieved nearly universal acceptance [15-17]. This is the Gene Ontology vocabulary. It was recently created by a consortium of groups representing many of the major research organizations and organisms. The breadth of representation in the consortium is probably a significant reason for its success. By including input from experts on a wide variety of organisms (i.e.: humans, mice, fruit flies, plants, yeast) they have created a vocabulary of gene function that is not biased or specific to any single organism, but rather applicable to all known species.

The vocabulary is represented as a directed acyclic graph with three branches: molecular function, cellular compartment, and biological process. Molecular function describes the specific job performed by the gene product in the cell. Biological process describes the broader cellular processes which the gene product or protein is involved in, such as signal transduction. Cellular component is used to describe the subcellular structures that the gene product is localized to. Because the ontology is represented as a directed acyclic graph, each term in the vocabulary has a unique ID number, but may have more than one parent term. A number of web-accessible browsers allow one to explore the various levels and branches of each tree. One of the best is the AmiGO browser at http://www.godatabase.org/cgi-bin/go.cgi.


AmiGO : GO:0003823 details - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Address http://godatabase.org/cgi-bin/go.cgi?query=GO:0003823&view=details&search_constraint=terms&depth
AmiGO
Get this GO term as RDF XML.
Get this data as a GO flat file.

antibody

Accession: GO:0003823
Synonyms:
immunoglobulin
Definition: None.
Term Lineage Graph view.

- GO:0003673 : Gene_Ontology (33650)
  - GO:0003674 : molecular_function (23707)
    - GO:0003793 : defense/immunity protein (522)
      - **GO:0003823 : antibody (147)**
    - GO:0004871 : signal transducer (2805)
      - GO:0004872 : receptor (1605)
        - GO:0004888 : transmembrane receptor (1326)
          - GO:0004892 : B-cell receptor (147)
            - **GO:0003823 : antibody (147)**

Figure 2: Screen-shot from AmiGO showing Gene Ontology graph for the term 'antibody' in the molecular function axis. As the ontology is represented as a directed acyclic graph, a single term can have more than one parent. In this case, antibody is a sub-classification of both 'defense/immunity protein' and 'signal transducer'.

The result of annotating genes with a structured vocabulary is that functional annotation data from different sources of can be readily integrated into a single repository. This represents a very high degree of information integration and promotes optimal data accessibility [18].

2.5 Microarray Sequence Annotation Resources

With the current popularity of microarray research, it is inevitable that other groups have been confronting the same question: how can we most effectively utilize existing annotation data for analysis of microarray gene expression data? Generally this involves acquiring and managing the available annotation information for each sequence on a microarray slide. There are several existing, web-based tools that have automated the process of acquiring gene annotations for a list of sequences [19-22]. These include:

- **Stanford Online Universal Resource for Clones and EST's (SOURCE)**

  http://genome-www5.stanford.edu/cgi-bin/SMD/source/sourceSearch
- **CloneUpdater**

  http://www.dbi.tju.edu/cloneupdater/html/template.php
- **Onto-Express**

  http://vortex.cs.wayne.edu/Projects.html
- **Array Organizing Tool (Arrogant)**

  http://lethargy.swmed.edu/index.asp

All of these tools function in a similar way. The user uploads a file containing a list of all of the sequences of interest, which may be represented by accession numbers or Clone ID's, and the program returns available annotations for each sequence. The tools differ somewhat in the source and types of annotations that they return.

One drawback to this approach is that there is no information on how the annotations are retrieved; particularly regarding the source of the information and its' age. Is it retrieved directly from the NCBI, or from a copy of the NCBI datasets installed in a local warehouse. If the data is from a local repository, it is important to ask how current is the data they are using? The NCBI resources are updated almost continuously, but data from a local repository may be any number of months old. Without meta-data describing the source and date of the annotation information, the user is left to gamble that what they receive is complete, accurate and up-to-date.

2.6 Project Setting

At OHSU, the Department of Pediatrics Biotechnology Core Facility serves as a local core lab for functional genomics and proteomics. It is located in the laboratory of Dr. Srinivassa Nagalla, and is available to OHSU researchers interested in performing microarray analysis on tissue samples from humans or mice. This facility uses "home-grown" microarrays which are printed on-site from libraries of cDNA clones. One advantage of this approach is that it is relatively easy to modify the composition and layout of sequences on the slide. The disadvantage is that this then requires very attentive data management to continually and accurately catalogue the layout of all slides. In addition, the home-grown approach creates significant need for flexible annotation tools that can respond to changing information needs. To this end, a dedicated team of students and specialists exists to provide informatics resources and technical support for researchers using the core facility. As his Medical Informatics Masters Thesis at OHSU, Brian Olmstead created the current warehouse for storing all of the experimental data.

Another important project is the Public Database Integration Project (PDIP), started by Mark Turner. The objective of the PDIP is to produce a central repository of structured and integrated gene annotation information that has come from a variety of external (public database) sources. The repository will then serve as a knowledge base for researchers conducting microarray experiments. The Public Database Integration Project is the direct precursor to this proposed thesis project. The goal is to build upon the work that was started, and complete the process of creating an effective gene function information resource for researchers conducting microarray gene expression experiments.

2.7 Specific Project Aims

The goal of this thesis project is to improve and optimize the functional analysis tools and resources available to researchers using the Biotechnology Core Facility. In order to accomplish this task, there are three major components:

1. Finish developing the PDIP data warehouse, to promote maximum quantity and quality of available gene annotation information. This includes identifying, obtaining and loading relevant sources of gene

annotations, integrating the sources into a single resource, and performing data quality control assessments.

2. Create and optimize methods for delivering gene functional annotation information to core facility users. Survey researchers to determine information needs and preferences, and respond with tools that accommodate research needs.
3. Improve high-level analysis options by implementing and demonstrating a tool, such as GenMAPP, for functional analysis and/or data visualization according to higher-level biological processes such as signalling cascades or regulatory mechanisms.

The first two components are intended to produce a set of informatics resources which will be used by researchers to answer questions regarding gene function throughout the experimental process. The third component attempts to improve the process of data analysis and presentation by providing tools to visualize data according to higher-level biological processes. When reporting results from microarray experiments, it is important to try to move beyond a simple list of up-regulated and down-regulated genes, and instead attempt to display those genes as part of a significant cellular process.

In addition to producing the above three components, it is also necessary to evaluate their effectiveness in the facility. The evaluation will be done in an on-going qualitative manner. Researchers will be asked how their information needs could be more effectively met. As tools are designed and made available, users of the new tools will be asked to provide feedback regarding their effectiveness, and ease-of-use. This feedback will be used to eavaluate the results of the project, and determine if it has achieved the goal of improving microarray data analysis according to gene function. It will also be incoporated into successive iterations of project results, so that the final components delivered will represent a high level of adherence to user needs.

3. Methods

The following section will describe the methods used in each of the project components: construction of the data warehouse of gene functional annotations, delivery of the information to the researcher, and the development of a pathway analysis resource.

3.1 Data Warehouse

The first component is the development of the data warehouse of gene annotations. This was accomplished as an iterative process of creating and modifying the warehouse table structures, identifying and obtaining important sources of sequence annotations, then parsing and loading the data sets while continually troubleshooting issues of error and data format incompatibilities.

It is necessary to briefly describe the initial state of the Public Database Integration Project prior to my taking it on as this thesis project. The warehouse had already been created and implemented with an Oracle relational database management system. In addition, many elements of the current table structure and the UniGene and LocusLink data sets had already been established. However, there was not yet a standard scheme for describing the function of gene products, when known.

Early efforts in the project explored other strategies for annotation, including linking genes with relevant MeSH terms based on literature citations. This involved downloading abstracts from PubMed, attempting to parse out genes that were cited within the abstract, and then linking them with classes of MeSH terms relating to gene function and disease. This approach was both labor intensive and resulted in a very low accuracy rate. Genes are usually cited by a three or four character symbol or a name, which may be several words long. In addition, a single gene may be identified by multiple different symbols, reflecting the evolution of research from multiple different labs. This makes it very difficult to reliably parse out all citations of genes within an abstract. Finally, even when all cited genes are reliably parsed out, the diverse nature of scientific publication still leads to low accuracy of correlation between genes mentioned and MeSH terms assigned to the publication.

The decision was made to re-structure the data warehouse and focus exclusively on annotating genes with terms from the Gene Ontology vocabulary. The GO vocabulary has clearly emerged as the de facto standard vocabulary for describing the function of eukaryotic genes [10], and it is used by expert authorities across multiple organisms. Several very large expert-curated data sets of GO annotations are publicly available, and the use of a shared vocabulary allows them all to be seamlessly integrated into a common repository. Thus switching to the Gene Ontology as the language for functional annotations resulted in a warehouse that was more accurate and required less labor to maintain.

The following data sets of gene annotations were selected and integrated into the warehouse: NCBI UniGene and LocusLink (for human and mouse), the Mouse Genome Informatics (MGI) file of annotations available through the Gene Ontology website, and the Japanese Riken FANOM (Functional Annotation of Mouse) database. The subsequent paragraphs will briefly describe the methods involved in obtaining each of these data sets.

The NCBI UniGene and LocusLink data sets are the core of the data warehouse. UniGene contains all of the GenBank sequence accession numbers organized by clusters which are believed to represent distinct genes. LocusLink maintains additional information for each cluster including the official and alternate symbol and names for the gene, the chromosomal location, functional annotations when available and links to other web resources. The data sets are available by ftp as zipped, tab-delimited text files. Mark Turner created scripts to parse out the desired elements and transform the data into the format necessary for loading into the data warehouse. Loading the data sets involves running the multiple scripts in a defined order. Since the NCBI data sets are still very dynamic, (they are updated almost continuously) it is important to keep the data warehouse as current as possible by frequently re-installing the most recent version of the data. As a general rule this is done at least every two to three months. Consequently it is also important to develop and maintain detailed and accurate protocols for computing and documenting the updates.

The Mouse Genome Informatics group at the Jackson Labs also produces a database of functional annotations for mouse genes, using the GO vocabulary [23]. The group uses its' own unique identifiers known as MGI accession numbers to differentiate genes. The LocusLink database maintains the associations between MGI accession number and UniGene cluster ID, providing effective integration of the two resources. The MGI database is available as a tab-delimited text file from either the Gene Ontology web site or the MGI site.

Since the file requires only mimimal modifications to be formatted for loading into the data warehouse, there have not been scripts written to automate the task.

Instead the modifications (selecting, re-naming, and re-ordering the relevant columns) are done 'manually' in the Excel spreadsheet program. The modified file is then loaded into a special table in the warehouse which contains only the Gene Ontology annotations. The MGI file is also updated weekly, but does not grow quite as fast as the NCBI resources. In order to keep the warehouse current, it should be updated at least every four to six months.

The final component of the warehouse is the Riken FANTOM data. The dataset was created as the result of a single large-scale international meeting with the goal of functionally annotating 21,076 fully sequenced mouse cDNA clones from their libraries [14, 24]. The dataset has significant overlap with the MGI data, but has enough additional information to merit its' inclusion in the warehouse. In order to integrate the FANTOM data into the warehouse, significant modifications must be made. The data set is released in a flat-file format, and must be reverse-engineered to the relational structure of the data warehouse. A program consisting of a series of SQL commands was written to parse out each type of data element into views which could then be merged into a single relational table. The FANTOM data set has not yet been updated since its' release, so there is no current need to update the warehouse.

3.2 Annotation Delivery

Once the dataset of gene annotations has been compiled, the next major question is how to most effectively deliver the information to the researcher who needs it. In order to answer this question it is important to characterize, in as much detail as possible, the information needs of the researchers, and the temporal patterning of those needs with respect to the workflow of the experiment.

A series of informal interviews with individuals involved in microarray experiments in the facility has yielded several generalizations.

1. Microarray experiments are generally done as a team effort, with multiple individuals involved in the experimental design and analysis processes. Information needs frequently manifest in a technician or student being sent to address questions posed by the principle investigator or other senior researcher.
2. Researchers tend to have questions about sequences and annotations at many points throughout the experiment. Information needs arise in the early stages of experiment design, and continue throughout the entire analysis process, possibly extending well after results have been published.
3. Researchers are frequently interested in a specific collection of genes or sequences prior to initiating the experiment, and want an easy means of determining whether or not those sequences are even represented on the microarray chip.
4. The user will likely desire multiple types of information relating to a gene or sequence of interest. These include the gene symbol, gene name, alternate names and symbols for that gene, the chromosomal location, the Gene Ontologies descriptors at multiple levels of detail, and other published research describing known interactions of that gene.

These generalizations can then be used to guide the development of systems for providing the desired annotation information. In terms of the functional implementation, three major possibilities exist: the use of spreadsheet files to deliver sets of annotations for a particular gene chip, analysis software that can incorporate and utilize annotation information, and a query able interface to the database of annotation information. Each of these has distinct strengths and weaknesses that will be described in the following paragraphs. In reality, it is unlikely that a single solution will satisfy all user information needs. The most ideal solution would implement two or more of these possibilities, and embrace the slight redundancy of complementary resources.

Spreadsheet File

It has been suggested that many of a users needs for sequence annotations could be addressed by simply providing them with a spreadsheet file, such as Microsoft Excel format, which lists all of the sequences on their microarray chip and the available information for each one. These files contain the accession number, gene symbol, gene name, chromosomal information and GO annotations for each spot on the microarray chip. The researcher would then search that file for any information they need. The advantage of this solution is that it is very easy to generate the file from a single query of the data warehouse. The disadvantage is that it is not ideal from the researchers perspective. Microsoft Excel was not designed to support complicated search queries of its spreadsheets.

Software

It is ideal to have annotation information available during the process of analyzing gene expression data. Several software programs for microarray data analysis now allow the researcher to directly import sequence annotations into the analysis procedure. These programs include Applied Mathematics GeneMaths, GeneSpring by Silicon Genetics, and the OmniViz Gene Expression Analysis package, among others. These three packages all use similar types of clustering algorithms to analyze data by creating groups of sequences that have similar patterns of expression across the experiment. The programs differ substantially in terms of the means of importing the annotations, the types of information that can be imported, and the ability to direct the data analysis according to annotation information. Currently GeneMaths and OmniViz are both in use in the Pediatrics Department Core facility. Both packages allow users to connect to an Oracle data warehouse and use scripts to retrieve functional annotations for the sequences in their expression data set. The annotations can then be used as additional variables for describing and categorizinging experimental data.

Query Interface

Another option for providing gene function annotations is to create a query interface to the data warehouse, so that researchers can ask specific questions about the sequences on the microarray chip used in their experiments. Processes of software development and high-level design were used to develop and implement a query interface. The following requirements for the user interface were determined:

- The interface needs to support queries relating to human and mouse genes.

- Users need to be able to determine which sequences and annotations are available on the microarray chip for their experiment.
- The interface needs to support multiple query types. Users may wish to search for genes based on an accession number, gene symbol, gene name, or Gene Ontologies descriptor.
- Additionally, the user must be able to request that the query return any of the above.
- The system should support running multiple queries from a single input; "batch" mode.
- Output should be in a format that can be printed out and/or saved to a file.

Regarding the implementation architecture, a web-accessible information retrieval program needs to include three components:

- An HTML form that captures the users request
- A component to process the request and direct it to the data warehouse
- A component to return the results to the users web browser.

The use Java Servlets with embedded SQLJ was chosen as the platform for deployment. Servlets are Java programs that run on a web server and can act as a middle layer between a request coming from a Web browser and a database or other application [25]. SQLJ allows Java programs to access information in a database using embedded SQL (Structured Query Language) statements [26]. This decision was based on the above components description, current technologies, and the broader bioinformatics goals of the Biotechnology Core Facility environment. Java Servlets are easy to develop and deploy, and SQLJ provides an efficient and powerful means of directing queries to the Oracle data warehouse and returning a result set.

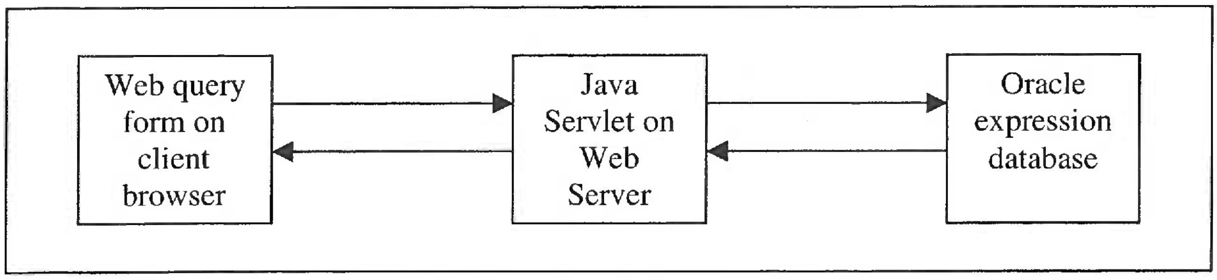

Web query form on client browser
Java Servlet on Web Server
Oracle expression database

Figure 3: Query Interface Architecture

3.3 Pathway Modeling

The next task is to explore means of using the gene annotations to produce a higher level of biological analysis. Microarray analysis results often describe sets of significant genes which are all elements of specific cellular processes, such as signalling pathways involved in cell cycle regulation or tumorigenesis [27-29]. It has been shown by microarray analysis that in mice, methamphetamine administration causes up-regulation of several members of the JNK/SAPK signalling pathway , which in turn regulates additional transciption factors[30, 31]. Typically these results are reported as a list of significant genes, which may be grouped according to their common functions. Pathway analysis could provide a more effective means of reporting such results.

Pathway analysis describes gene expression data by using a visual diagrams to depict important cellular and molecular biological processes. Expression data is displayed graphically as a a collection of related genes or a diagram of a metabolic or signalling pathway, using a color-scale to describe changes in expression-level for each component[32]. The result is a powerful and simple illustration displaying expression level changes for a set of genes which are all involved in a common biological process.

So far there have been very few attempts to do pathway analysis of microarray data in the Pediatrics Department Biotechnology Core Facility. This likely due to a perceived lack of resources, or lack of familiarity with publicly available resources. In order to promote this technique among researchers, it is necessary to find and obtain suitable resources and then become expert enough to aid others. This involves learning the software, optimizing it to the local environment, demonstrating its' utility and effectiveness, and finally, creating detailed protocols so that others can then use it in a time-effective manner.

The current leading and available resources for pathway analysis of microarray data are GeneSpring, a proprietary microarray data analysis package by SiliconGenetics, and GenMAPP, which is a publicly available program produced by a team of resarchers at the University of California San Francisco.

GeneSpring

The pathway mapping feature in GeneSpring is one component of a much larger microarray expression data analysis package. Essentially it allows a user to import a pre-existing image file of a pathway in .jpeg or .gif format. Then graphical tools are used to specify the regions in the image which correspond to specific genes on the microarray slides, creating a link to the actual data. Once this has been done, the program displays a series of color-coded blocks representing the relative expression level of that gene on each slide.

A major drawback of this approach is that many pre-existing pathway diagrams, such as those published by BioCarta, were not visually designed to allow incorporation of additional elements. The result becomes cluttered, crowded, and generally difficult to readily interpret. If the user were to create their own pathway models specifically designed for this purpose, the results would probably be much easier to read. However, one advantage of this method is that data from multiple experimental conditions can all be displayed at once.

Ultimately it was decided that GeneSpring would not be purchased for use in the Biotechnology Core Facility, so its further potential was not pursued.

GenMAPP

GenMAPP was designed for the sole purpose of displaying gene expression data according to known biological processes. The program has three main components: a MAPP file, which is a diagram showing a biological relationship among genes or gene products, a local database of gene ID's and annotations, and an expression dataset which contains the experimental data and user-defined rules for coloring gene products on the MAPP. Users create the MAPP file for their pathway of interest, and then link the elements to their expression data via the local database.

The program comes with a simple drafting toolbox for creating the MAPP files. The developers have strongly promoted the open-source philosophy of sharing MAPP's once they have been created. Consequently there is already a substantial and growing archive of signalling pathway MAPP files from a variety of sources. In addition, they have utilized the Gene Ontology vocabulary to create a set of MAPP files which represent all of the known genes of a particular GO classification, such as nuclear receptors.

The problem with the archived MAPP's, however, is that the GenMAPP developers have favored SWISS-PROT as the source for sequence identifying accession numbers. The Biotechnology Core Facility uses the American alternative, GenBank ID's and it is currently not possible to convert an already created MAPP file. Thus the existing archives of MAPP files can not currently be effectively used by local researchers, except perhaps as a template for generating an identical pathway file using GenBank accession numbers. However, there is a new version scheduled for release very soon, and the developers have promised additional support for GenBank and UniGene identifiers in the new release.

Despite the few (hopefully temporary) shortcomings, GenMAPP appears to be the best currently available tool for performing pathway analysis of microarray gene expression data, and consequently will be chosen for use in the Biotechnology Core Facility. In order to promote GenMAPP for use by OHSU researchers, it is necessary to become the local expert user, and develop a protocol for streamlining the process. It is also necessary to effectively demonstrate its' potential by performing some sample analyses for others. The result will be a tool that is optimized and supported for use in the core facility.

3.4 Methods Summary

The major components of this thesis project are the creation of a data warehouse of gene sequence annotations, the development of multiple tools to convey the available knowledge to researchers at the time of need, and a demonstration of and protocol for using the technique of pathway analysis on microarray expression data.

4. Results

4.1 Data Warehouse

The data warehouse now contains gene annotation data using the Gene Ontology vocabulary from NCBI's UniGene and LocusLink, the Jackson Labs MGI data set, and the Riken FANTOM data. The warehouse has been updated regularly to ensure that the data is current, and detailed protocols (see Appendix I.B) have been created so that others can continue with the updates in the future. The scripts for loading and updating the data continue to exist as a heterogenous combination of UNIX, PL-SQL, and Java formats. This is largely due to the evolving nature of the data warehouse, and a lack of time to devote to the task of streamlining the process. The protocols are intended to clarify any confusion and be specific enough to guide a user through the whole process. A basic familiarity with Excel, Unix and Oracle is assumed.

The contents of the warehouse are organized as a non-normalized relational structure (Appendix I.A). The central table for retrieving annotations is the gene_symbols table, which contains integrated annotation information for each UniGene cluster ID. The other important tables are the array_layouts table, which lists all of the sequences on each chip by GenBank accession number, and the gene_accession_numbers table, which lists all of the accession numbers assigned to each UniGene cluster ID. By linking these three tables through foreign key relationships, one can then retrieve annotation information for any sequences on the microarray chip, or any other sequences of interest. In order to simplify this process, a view named chip_annotations was created, which contains all of the sequences on each of the currently actively used microarray slide layouts (Appendix II.A). This view then serves as the warehouse interface to the data access tools developed for researchers. In addition, it can also be used to generate descriptive statistics summarizing the annotation information that is available for each of the slide sets; Figure 4 (page 32).

Chip Set	Total Spots	Distinct UniGene ID's	Gene Symbol (%)	GO Biological Process	GO molecular function	GO cell component
Human R66	12686	11137	7380 (58%)	3449 (27%)	3545 (28%)	2658 (21%)
Mouse BMAP	13293	10298	9882 (74%)	3511 (26%)	3855 (29%)	3034 (23%)
Mouse NIA	12513	10397	8884 (71%)	2834 (23%)	3110 (25%)	2505 (20%)

Figure 4A

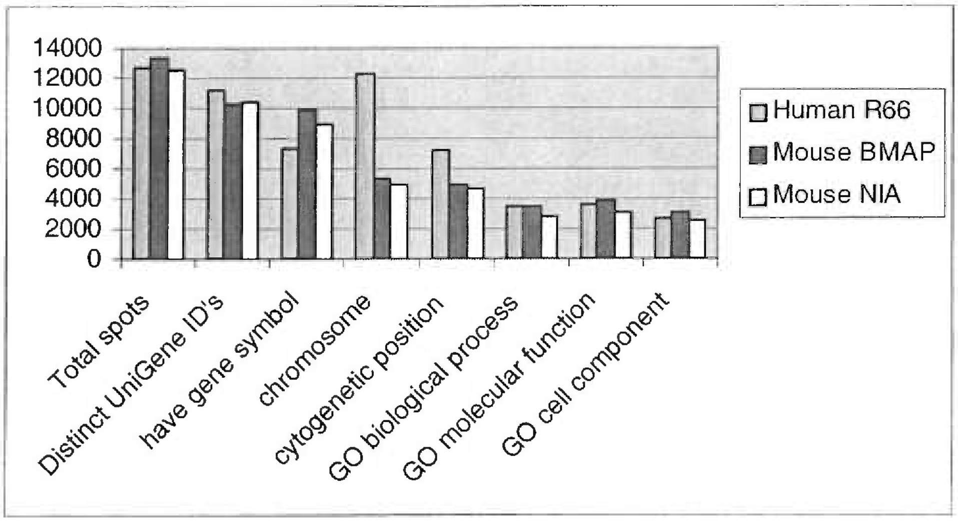


Figure 4B

Figure 4A,B: Table and chart showing each of the currently actively used slide sets and the count of sequences and available annotations for each one. The gene symbol category is used to represent the sequence as a known gene, as opposed to a sequence that is known to be expressed in the organism, but not yet classified as a specific gene.

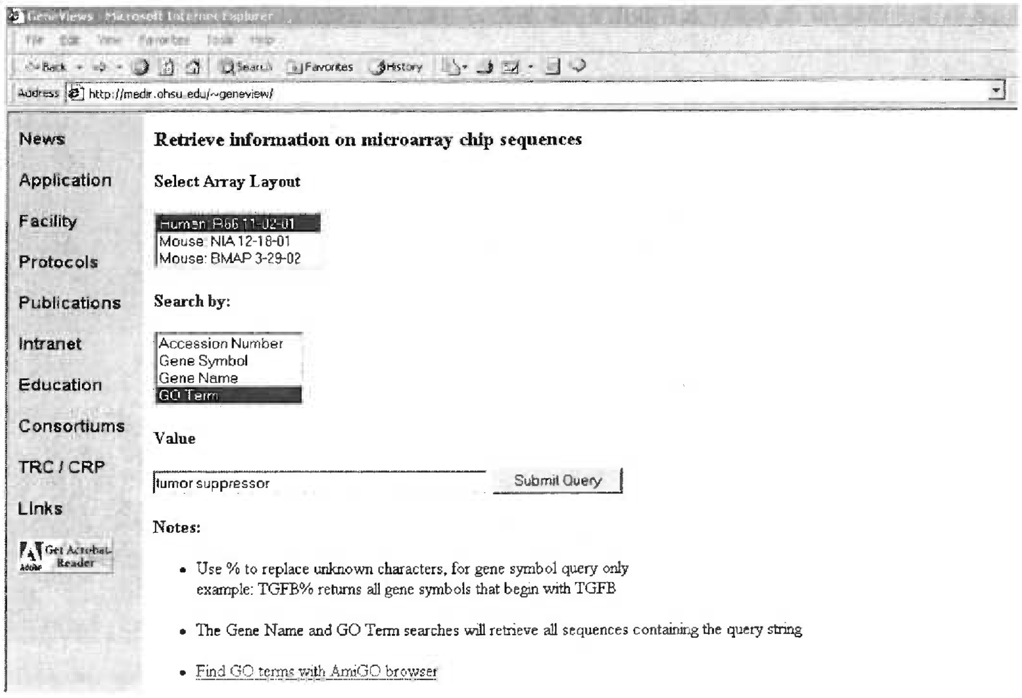
4.2 Annotation Delivery

Several tools for accessing sequence annotations have been made available in parallel to core facility users. The microarray data analysis programs GeneMaths and OmniViz support integrating the sequence annotations into the analysis process. For GeneMaths, Mark Turner developed an embedded macro which connects to the data warehouse and retrieves annotations for the sequences being analyzed. OmniViz also enables a direct connection to the data warehouse, where SQL commands can be used to return sequence annotations for the microarray slide layout used in the experiment. An OmniViz protocol has been written in order to enable novice users (Appendix II.B).

In addition, Microsoft Excel spreadsheet files have been created for each chip. These contain all of the sequence accession numbers on the slide, and the gene symbol, gene name, chromosomal locations, and GO terms at two levels in the hierarchy (the assigned term, and a 'broad' term which is more general). A link to the spreadsheet files will be implemented on the laboratory web site, 'Gene Views' (http://medir.ohsu.edu/~geneview/).

Finally, there is the QChip web-interface. Figure 5 shows screen shots of the query entry form, and a sample of returned results. The interface is accessible through the GeneViews home page from any work station on the OHSU campus. The current implementation supports most of the requirements described in the methods section. Users can search the three most commonly used arrays by accession number, gene symbol, gene name, or Gene Ontology descriptor. The accession number search retrieves all of the sequences on the chip that are in the same UniGene cluster as the accession number entered by the user. Thus it should retrieve sequences that correspond to the same gene as the value entered by the user. The gene name and GO term searches both retrieve all sequences that contain the search string. The gene symbol search is precise, but the '%' symbol can be used as a wildcard operator to broaden the request.

The results are returned as an HTML table with links to the NCBI UniGene database for each sequence returned. From there, a user can readily link to LocusLink and access almost any other data available for that sequence. The query results can also be returned in a semi-colon delimited format that can be copied and converted to an Excel spreadsheet. At this time, the user interface only supports running a single query at a time. The batch function may be implemented in the future, if time permits.


GeneViews - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Address http://medir.ohsu.edu/~geneview/
News
Application
Facility
Protocols
Publications
Intranet
Education
Consortiums
TRC / CRP
Links
Get Acrobat Reader
Submit Query

Retrieve information on microarray chip sequences

Select Array Layout

Human: R66 11-02-01
Mouse: NIA 12-18-01
Mouse: BMAP 3-29-02

Search by:

Accession Number
Gene Symbol
Gene Name
GO Term

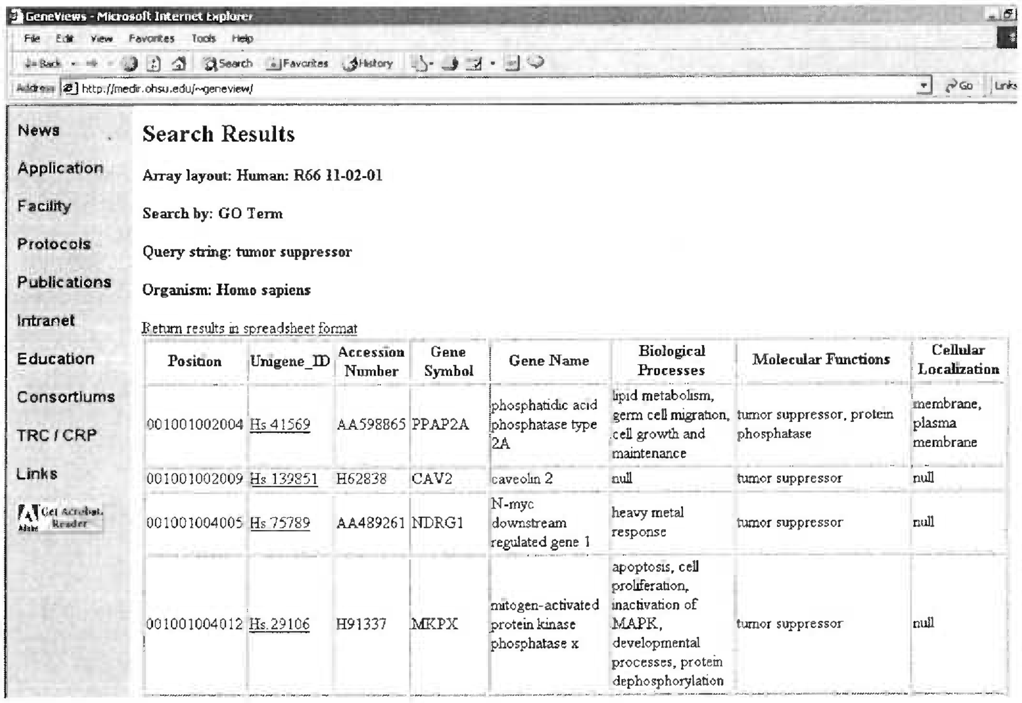
Value

tumor suppressor

Notes:

- Use % to replace unknown characters, for gene symbol query only
  example: TGFB% returns all gene symbols that begin with TGFB
- The Gene Name and GO Term searches will retrieve all sequences containing the query string
- Find GO terms with AmiGO browser

Figure 5A: QChip query interface


GeneViews - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites History
Address http://medir.ohsu.edu/~geneview/
Go Links
News
Application
Facility
Protocols
Publications
Intranet
Education
Consortiums
TRC / CRP
Links
Get Acrobat Reader

Search Results

Array layout: Human: R66 11-02-01

Search by: GO Term

Query string: tumor suppressor

Organism: Homo sapiens

Return results in spreadsheet format

Position	Unigene_ID	Accession Number	Gene Symbol	Gene Name	Biological Processes	Molecular Functions	Cellular Localization
001001002004	Hs.41569	AA598865	PPAP2A	phosphatidic acid phosphatase type 2A	lipid metabolism, germ cell migration, cell growth and maintenance	tumor suppressor, protein phosphatase	membrane, plasma membrane
001001002009	Hs.139851	H62838	CAV2	caveolin 2	null	tumor suppressor	null
001001004005	Hs.75789	AA489261	NDRG1	N-myc downstream regulated gene 1	heavy metal response	tumor suppressor	null
001001004012	Hs.29106	H91337	MKPX	mitogen-activated protein kinase phosphatase x	apoptosis, cell proliferation, inactivation of MAPK, developmental processes, protein dephosphorylation	tumor suppressor	null

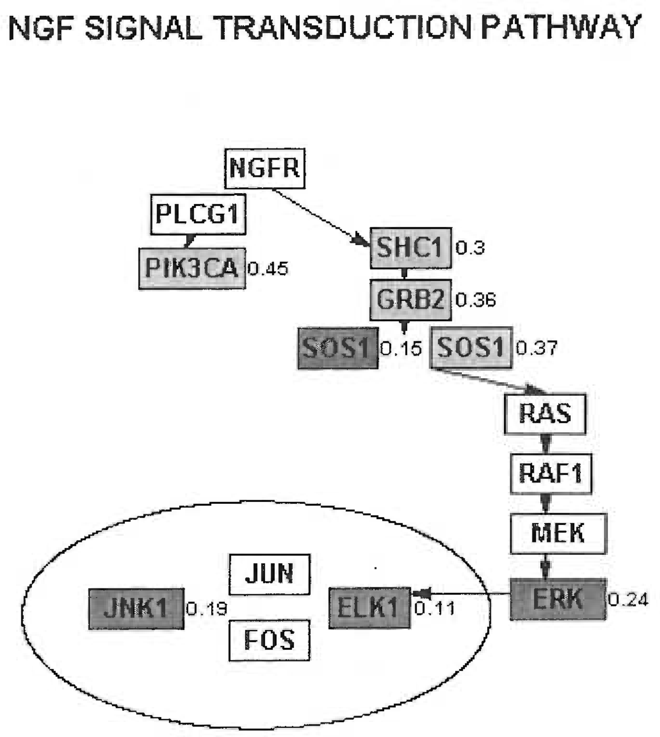
Figure 5B: Sample results output

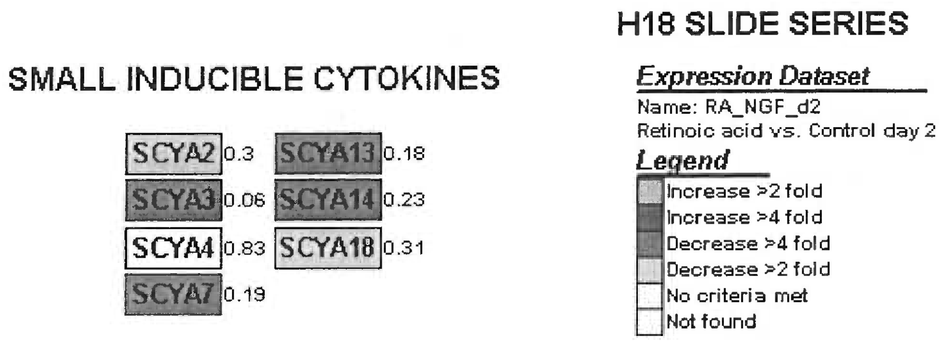
4.3 Pathway Modelling

The GenMAPP software has been used to produce a demonstration of its' potential. The program supplies graphical tools that can be used to create a MAPP file of the pathway or gene collection of interest. Additionally there is a new tool that will automatically create the MAPP file from an input list of accession numbers.

Figure 6 shows a sample analysis using the NGF pathway and data from an experiment of retinoic acid applied to cell lines. This sample diagram is not intended to represent actual scientific results, only as a model of the type of analysis and data visualization that the program can be used for. Appendix II.C contains a protocol for using the program in the local facility where sequences are identified by GenBank accession numbers.

4.A





4.B

Figure 4: Sample GenMAPP analysis. Each box represents a gene on the microarray slide. The legend describes the fold change for each color. 4.A shows a representation of the nerve growth factor signalling pathway as described in the BioCarta database (http://www.biocarta.com/pathfiles/ngfPathway.asp). 4.B is a collection of genes representing a common function, the small inducible cytokines.

5. Evaluation & Discussion

This section will focus first on evaluating each of the project components and then will consider the potential life-span of the project by placing it within the context of new developments in the broader field of bioinformatics.

5.1 Evaluation of Project Components

It is necessary to evaluate the delivered results in order to determine if the tools succeeded in improving functional analysis resources. The goal of this thesis project was to develop information resources for researchers performing microarray gene expression analysis in the Department of Pediatrics Biotechnology Core Facility. To accomplish this task there were three major components: develop a data warehouse of gene functional annotations for human and mouse sequences, provide researchers with easy-to-use access to the annotation information, and promote higher-level biological analysis by initiating and demonstrating a tool for visualizing expression data according to specific biological processes.

The small number of researchers performing functional analysis of microarray data at any given time makes it quite difficult to complete the development process and then generate an effective volume of evaluation information. This impediment was resolved by attempting to characterize information needs and desires as completely as possible during early stages of development, and then by continuously requesting and incorporating suggestions for improvements in each of the tools throughout the development process.

5.1.1 Data Warehouse

At the beginning of this thesis project, the Public Database Integration Project portion of the data warehouse contained the UniGene and LocusLink data for human and mouse. It also contained the Gene Ontology annotations which were available in LocusLink. At the time, there was an attempt to annotate gene products using MeSH terms from the National Library of Medicine. This proved to be significantly labor-intensive and produced results of low accuracy, and dubious reliability.

The decision was made to switch to the Gene Ontology classification for describing gene products. This decision allowed the incorporation and integration of significant additional amounts of annotation data, specifically the Jackson MGI and Riken FANTOM annotations data sets. It also improved the accuracy of the data, as the incorporated data sets come from expert authorities, and are regularly updated.

As annotation data was rapidly being produced over the one year interval of the project, the warehouse continued to grow significantly each time one of the component data sets were re-loaded. Clearly, the content and utility of the warehouse has evolved from the state it was in prior to being taken on as the focus of this thesis project. However, it should still be seen as a work in progress. In order for the warehouse to remain useful, it will be necessary for other lab members to continue keeping the component data sets up-to-date from the each of the sources. The detailed protocols that I have developed will be instrumental in this task.

5.1.2 Annotation Delivery Tools

Currently, the tools for accessing and using chip sequence annotations for functional analysis are the QChip web interface, the sofware packages GeneMaths and OmniViz, and the spreadsheet files that have been developed for each slide set. Researchers using the Biotechnology Core Facility have expressed significant interest in and used each of these tools.

The spreadsheet files provide information that is equivalent to the output of other available resources (CloneUpdater [33], Stanford SOURCE [19]) for providing annotations of microarray slide sequences. By creating the file and providing a link to it from the facility web site, future users are spared the time expense of having to locate and learn those resources on their own. The development of a protocol for using OmniViz to perform cluster analysis using annotated sequences has been of substantial interest to researchers as well.

In order to ensure user satisfaction, the QChip interface was created as an iterative process of development, and response to feedback. The improvements based on feedback included the incorporation of hyperlinks to NCBI resources, the implementation of case-insensitive queries for gene symbols, and the implementation of semi-colon delimited output format for exporting results to a spreadsheet.

A major advantage of the QChip interface over other publicly-available resources is that only QChip provides the ability to specifically query the set of sequences on the slide sets in the local facility. This is particularly important since the slides are made on-site and each contain 12,000 – 13,000 sequences. When additional new slide sets become available in the future, the program and the data warehouse are designed to be easy to update and incorporate the new slides.

One resource that users have requested that remains to be implemented is the "genes of interest" table. Researchers have requested the ability to readily create and maintain a list or spreadsheet file of a set of genes on the microarray slide that they are specifically interested in tracking throughout their experiment. Currently, the QChip query interface can be used to find the genes on the slide, and the results can be returned in a spreadhseet format that can be copied into a single file. This is still a rather awkward and time-consuming task, and a tool that could readily automate the process would be beneficial. A number of strategies have been discussed for implementing this procedure, but a lack of time has so far hindered its completion.

5.1.3 Pathway Modelling

The task of initializing and using the GenMAPP software was reasonably straightforward, however, it was difficult to produce a meaningful demonstration in this project setting. The initial goal was to investigate intracellular signalling pathways as candidates for visualization analysis of microarray data.

There were multiple problems with this approach. A major source of difficulty was the general composition of the microarray slides used in the local facility. As it was shown in Figure 3, for each of the slide sets, only 20-30% of the sequences on the slide represent known and characterized genes, such as signalling pathway elements. With experimental data, this proportion tends to remain constant across significantly differentially expressed genes: the majority of them are EST's with very little biological characterization. Thus they can not yet readily be assigned a role in the experimental condition. Another result of the slide compositions is that for several common signalling pathways (ie: NGF, IGF, TGFB) it was determined that only some portion of the pathway components were represented by sequences on any of the available microarray slides. This meant that any generated pathway representation would be missing data for multiple elements.

Another major complication was that based on the available experimental data, signalling pathway elements do not appear to be significantly differentially expressed in recent studies conducted in the Department of Pediatrics Biotechnology Core Facility. Thus it proved very difficult to produce a meaningful demonstration of the potential of this type of analysis in the local setting. The delivered result is a protocol for using the GenMAPP software in the local facility, and a sample demonstration that does not represent scientifically meaningful results.

Despite these obstacles, I still believe that this type of systems biology analysis will ultimately prove to be important in the near future. As microarray data continues to accumulate in public repositories, it will soon be possible to mine data from multiple different experiment sets and begin to generate a larger picture of the types of genes that are most likely to be differentially expressed across varied experimental conditions. As this ability develops, it should become easier to determine the types of data that would be ideal for presentation with this method.

5.2 Developments in Bioinformatics

Over the time course of this project, the broader discipline of bioinformatics continued to progress and developments were made in several areas that relate to functional analysis of microarray data. It is important to discuss these developments and consider how they might impact this project in the near future.

5.2.1 Gene Annotations

A researcher using any annotation information resource to analyze microarray data needs to believe that they have all of the information currently available. The global plan to annotate the entirety of the human and mouse genomes is well underway, and the information available changes and grows at a significant rate. Thus, staying caught up with available sequence annotations is a process of continuously chasing a moving target. In order to keep a local warehouse up-to-date, it is necessary to frequently acquire and re-load the latest version of the available data sets. This is not a particularly efficient way to accomplish the result. Like the genome sequencing project, at some point in the future the annotation process will be completed as well. When the rate of new information levels off, it will be less necessary to dedicate resources to the task of continually chasing the latest data. Achieving this goal, however, is still at least a couple of years in the future.

In addition, it remains to be determined what the ideal level of granularity is for sequence annotations data that are provided to the microarray researcher. What is the "right" amount of information that accomodates researcher needs without over-whelming them with far too many details?

5.2.2 Distributed Annotation Systems

The need to integrate annotation data from disparate sources has been widely acknowledged recently, and significant progress is being made in developing better tools to automate that process using the internet. The Distributed Annotation System (DAS or BioDAS) promises to revolutionize the process of collecting and integrating genome sequence annotations from multiple sources [34, 35]. BioDAS uses XML-based data definition standards, and end-user clients which are configured to integrate data from multiple different DAS servers.

DAS is just beginning to achieve signficant implementation in the bioinformatics community. It is currently in use on an experimental basis by the organism-specific genome groups: EMBL-Ensemble (human) WormBase, and the Berkley Drosophila Genome Project. Presently it is designed to provide annotations only according to genomic landmarks (ie: chromosomal position), and not functional annotations of sequences by accession number as used in microarray studies. This goal, however, is easy to visualize: the ability to receive a user-customized set of the latest available gene annotation data from which ever sources are desired for any set of sequence accession numbers. When this is fully realized, there may no longer by any need to maintain a local data warehouse.

5.2.3 Pathway Modelling

Using models of biological processes or pathways to display microarray data is another technique that will continue to develop in the near future. Two inter-connected areas of research that will drive advances in this technique are the development of publicly available databases of known biological and biochemical pathways, and progress towards a standard graphical language for pathway depiction. Presently, there are several publicly accessible repositories of metabolic and signaling pathways in various organisms; including the KEGG database [36] and BioCarta. These databases are still very new, and currently exist in several heterogeneous formats, which effectively prevent interchange of data from one resource to another. There is not yet any consensus on the best way to store pathway representations (i.e. as a single image file or as a map of interconnected components) such that they are of maximal utility to the researchers interested in them. Neither is there a standard graphical language for creating pathway representations. A standard format for pathway display and interchange, and the development of an authoritative repository of pathway information would both go far in driving this type of analysis for microarray data. The result would be a much more sophisticated biological interpretation of the gene expression profiles generated by each experiment.

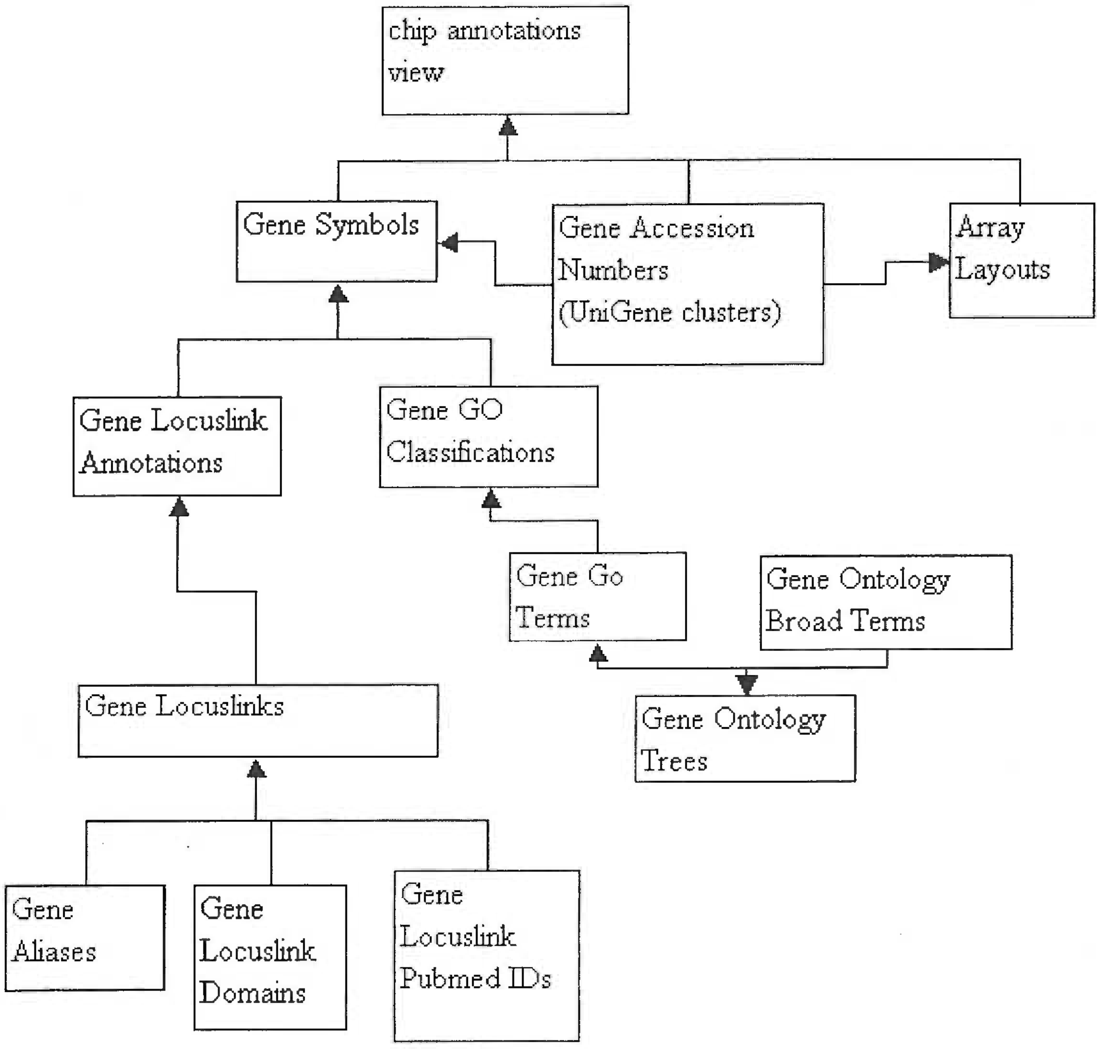
5.3 Project Conclusion

The delivered components of this thesis project include the improved data warehouse, new tools for accessing sequence annotations, an introduction to pathway mapping possibilities, and protocols for use and maintenance of each item. Together, these form a new resource for functional analysis of microarray data to be used by scientists in the Department of Pediatrics Biotechnology Center. Initial responses to the tools have been positive, and significant attempts have been made to incorporate additional requests.

Continued developments in bioinformatics will necessarily render these tools obsolete at some point in the future. However, at the present time, the combined tools and protocols provide a new and effective resource that is in line with the current state of functional analysis of microarray data by other groups nationally and internationally. Researchers now have multiple means of accessing functional annotations of microarray slide sequences and for incorporating the annotations into the analysis process. In addition, the modeling tool, GenMAPP provides a new resource for visualizing and displaying microarray data as a clear account of biological event in the organism of interest. These tools will be used to conduct microarray gene expression analysis, which will in turn promote the greater mission of fighting human disease at Oregon Health & Science University and in the global biomedical research community.

Appendix 1.A

PDIP Warehouse Schema



This schema represents the tables comprising the Public Database Integration Project components of the Oracle data warehouse. Arrows indicate the direction of data "flow"; which is implemented either primary key/foreign key relationships or through data replication across tables.

Appendix 1.B

Downloading UniGene and LocusLink databases.

The UniGene and LocusLink databases are the heart of the data_integration collection of sequence annotations. The data is updated almost continuously by NCBI. In order to keep the data warehouse as current as possible, these data sets should be re-loaded at least every 3-4 months.

1. **Download files**. Data sets are located at:
   ftp://ncbi.nlm.nih.gov/repository/UniGene/
   ftp://ncbi.nlm.nih.gov/refseq/LocusLink/
   For UniGene, the files for human and mouse sequences are named **Hs.data.z**, and **Mm.data.z**, respectively. The LocusLink files are **LL_tmpl** and **Loc2cit**. Download the files and use WinZip to uncompress the UniGene files.
   Save the files in data folder on Turgidson (/usr/local/dbload/)

2. **Parse data sets**. Mark has written UNIX programs to parse the files for loading into the Oracle database. The parsing programs are: **parse_unigene** for the human data and **parse_unigene_mouse** for the mouse data. (located at: /usr/local/dbload/bin/). In addition, a second set of programs parse out the information relating to the corresponding proteins (**parse_unigene_prot**, and **parse_unigene_prot_mouse**). It is necessary to run the parsing programs within the same folder that the data files are in. The parsing programs create a series output files; one containing accession numbers, and the other containing the associated gene symbols. The output files are **h_unigene_acc.ctl**, **h_unigene_sym.ctl,** and **h_unigene_prot.ctl** (mouse files replace the 'h' with 'm')
   Do the same with the LocusLink data, the parsing programs are named **parse_locuslink** and **parse_loc2cit**. The output files all have the .ctl extension.

3. **Remove old tables from Oracle db.** Once the data files have been parsed, they are ready to be loaded into the Oracle database. The loading program will run much faster if you manually truncate the corresponding tables in Oracle (SQL*Plus), rather than letting the loader program do it.
   ```
   TRUNCATE TABLE gene_accession_numbers
                  gene_symbols
   ```

4. **Load new data**. It is necessary to load the human files first, as the human .ctl file replaces all the data currently in the table (the mouse file then appends to the new data.)
   The program **run_sqlldr** contains the login information for Oracle.
   Load the files in this order:

   h_unigene_sym.ctl
   h_unigene_acc.ctl

h_unigene_prot.ctl
m_unigene_sym.ctl
m_unigene_acc.ctl
m_unigene_prot.ctl
ll_aliases.ctl
ll_annotations.ctl
ll_domains.ctl
ll_summary.ctl
loc2cit.ctl

After loading each data file, check the results of the process. The program generates output files with the extensions **.log** and **.bad** which contain information on rows that failed to load. Look at these files to determine that all or most of the rows loaded properly. If there are large numbers of errors, there has probably been a change in the format of the data set. It will be necessary to change either the data, or the specifications in the warehouse.

5. **Organize warehouse**. Once all the files have been loaded it is necessary to run additional SQL programs which copy LocusLink and GO annotation data back into the gene_symbols table.
   The programs are:
   move_locuslink_to_symbols.sql
   add_gene_function.sql
   update_resgen_with_new_unigene.sql
   MouseGo

   (located at: /usr/local/dbload/sql/)

   They need to be run in the above order.
   In Oracle: **START move_locuslink_to_symbols.sql**

   Update_resgen and MouseGo are PLSQL programs, which need to be run with the "CALL" statement.
   (CALL update_resgen_with_unigene)

   MouseGo is JAVA based and requires that you declare a variable that will be passed as an argument.

   ```
   VARIABLE  a varchar2(30);
   CALL mousegoprj.main(:a);
   ```

6. **Check results**. After loading data and running update programs, it is a good idea to check the results by running some sample queries to make sure the data seems to be as expected. Check content in the gene_symbols table for both human and mouse sequences.

Appendix 1.C

Loading the Jax MGI data set

1. **Download file**
   ftp://www.informatics.jax.org/pub/informatics/reports/gene_association.mgi
   (currently version 1.7)

2. **Organize for loading into warehouse**
   Save the file, and open as an Excel spreadsheet. Add column headings for:
   mgi_accession_number, gene_symbol, gene_name, alternate_symbols, GO_accession_number, GO_type, organism, source

   Fill in 'mouse' for organism and 'Jax MGI' for source.
   Change GO_type F,C,P values to molecular function, cellular component, biological process.
   Change mgi_accession_number column to numeric format by removing the "MGI:" string.
   Save the file in comma delimited .csv format.

3. **Load into warehouse**
   Copy the .csv file into the dbload folder on Turgidson. (/usr/local/dbload)
   Then use the following UNIX commands to sort, remove duplicate rows and covert from DOS format:
   sort –u old_file | fromdos > new_file

   Use **bin/run_sqlldr** for loading data into the data integration section of Oracle.
   The control file is **GO_associations_table.ctl**
   Check the control file to ensure that name of data file is correct, and columns to load are correct.
   After loading, check the .log and .bad files to ensure that data loads correctly.

4. **Copy annotations into gene_symbols table.**
   Finally, run an SQLJ program which retrieves the annotation descriptions, and copies them into the gene_symbols table. MouseGoPrj/AddMouseGo. To do this in Oracle, it is first necessary to declare a variable: **variable a varchar2(20);**
   Then call the program, and pass it the empty variable with: **call mousegoprj.main(:a)**

5. **Check results.**
   Run sample queries to make sure the data has been correctly loaded, and copied into the gene_symbols table.

Appendix 1.D Riken FANTOM data

The Riken group is the Japanese equivalent to the NCBI in the U.S. and the EBI in Europe. The goal of the FANTOM project is to functionally annotate the entire mouse genome (genes identified by RIKEN Mouse Gene Encyclopaedia project). The project involves significant collaboration with MGI and uses the Gene Ontology vocabulary.

The Riken FANTOM data is available in a flat file format. In order to be integrated into the warehouse, it needs to be coverted into a relational table. First load the data as it is into a temporary table, then use the SQL program to parse it and copy it into the gene_go_classifications table.

Loading the Riken FANTOM data set

1. **Download file**
ftp://fantom.gsc.riken.go.jp/ The file is **anndata.txt.gz**. Use WinZip to decompress, and open into an Excel spreadsheet. Check to see that the data has the following column headings in this order: **masterid, mgi_accession_number, qualifier, anntext, datasrc, srckey, evidence**.
Then save as a tab-delimited .txt file.

2. **Load into warehouse**
Copy the .txt file into the dbload folder on Turgidson (/usr/local/dbload). Then use **bin/run_sqlldr** to load the data into the temp table in Oracle. The control file is **temp_fantom_data.ctl**. Make sure the path in the control file is correct.

3. **Convert the temp_fantom_data table to relational format for integrating with other annotations.**
The SQL **parse_fantom_data program** does this by parsing out data elements to create a series of relational views which can then be merged into a single table.

SQL parse_fantom_data

```
create or replace view f_gene_symbol as
select distinct mgi_accession_number, qualifier, anntext gene_symbol
from temp_fantom_data
where qualifier = 'gene_symbol'
and mgi_accession_number != '#N/A'

create or replace view f_gene_name as
select distinct mgi_accession_number, qualifier, anntext gene_name
from temp_fantom_data
where qualifier = 'riken_def'
and mgi_accession_number != '#N/A'
```

```
create or replace view f_gene_annotation as
select distinct mgi_accession_number, qualifier go_type, anntext,
go_accession_number
from temp_fantom_data
where qualifier like 'gene_ontology_%'
and mgi_accession_number != '#N/A'

create or replace view f_final as
select  a.mgi_accession_number, s.gene_symbol, n.gene_name,
a.go_accession_number, a.go_type, 'Mus musculus' organism, 'FANTOM'
source
from f_gene_symbol s,
f_gene_name n,
f_gene_annotation a
where   s.mgi_accession_number = n.mgi_accession_number
and   n.mgi_accession_number = a.mgi_accession_number

insert into gene_go_classifications
(mgi_accession_number,gene_symbol,gene_name,go_accession_number,
go_type,organism,source)
select distinct mgi_accession_number,gene_symbol,gene_name,
to_number(substr(go_accession_number,4)),
decode (go_type,
'gene_ontology_F', 'molecular function',
'gene_ontology_P', 'biological process',
'gene_ontology_C', 'cellular component'),
organism,source
from f_final
where length(gene_symbol)<=20 and length(gene_name)<=200
```

4. **Copy annotations into gene_symbols table.**
   Finally, run an SQLJ program which retrieves the annotation descriptions, and copies them into the gene_symbols table. MouseGoPrj/AddMouseGo. To do this in Oracle, it is first necessary to declare a variable: **variable a varchar2(20);**
   Then call the program, and pass it the empty variable with: **call mousegoprj.main(:a)**

5. **Check results.**

Run sample queries to make sure the data has been correctly loaded, and copied into the gene_symbols table.

Appendix II.A

chip_annotations view in data warehouse

There is a view set up in the data_integration section of the warehouse that contains annotations data for all of the sequences on each of the currently actively used microarray slide sets. This view is used by the QChip query interface, and can also be used to readily retrieve annotations data for import into OmniViz.

The SQL statement for creating the view is:

```
create or replace view chip_annotations as
select
        l.array_layout_name, l.accession_number, l.position,
        n.unigene_cluster_id, s.gene_symbol, s.gene_symbol_upper,
        s.gene_name, s.organism, s.chromosome, s.cytogenetic_position,
        s.broad_biological_process_list, s.biological_process_list,
        s.broad_function_list,s.function_list,
s.broad_cellular_component_list,
        s.cellular_component_list
from array_layouts l,
         gene_accession_numbers n,
         gene_symbols s
where l.accession_number = n.accession_number
        and n.unigene_cluster_id = s.unigene_cluster_id
        and (array_layout_name like 'Human: R66 11-02-01' or
             array_layout_name like 'Mouse: BMAP 3-29-02' or
             array_layout_name like 'Mouse: NIA 12-18-01')
```

When a new slide set becomes available it will be necessary to update this view to incorporate annotations for the new set. The accession numbers for the layout should have already been added to the array_layouts table. To update this view, simply add

array_layout_name like *'new layout name'*

to the final 'and' clause, then copy and run the statement in SQL*Plus.

Appendix II.B

Using OmniViz to analyze microarray expression data

OmniViz software provides a powerful means for visually integrating functional annotations with gene expression data, and can also include annotation categories in the analysis process. The core features of OmniViz are very sophisticated algorithms for data integration and clustering. Unlike most microarray analysis tools, OmniViz was initially developed as a broad spectrum informatics tool, and then applied to gene expression research.

OmniViz supports direct data import from ODBC databases. This makes it easy to load one's experiment consisting of the normalized microarray data from multiple slides, and then rapidly and simultaneously import the available annotations for all of the sequences on that chip. Then it is possible to apply clustering algorithms based on both the numeric data and annotation categories.

Loading annotation data into OmniViz

1. Prepare Excel spreadsheet of normalized data. The accession number column must be exactly titled 'ACCESSION_NUMBER'
   It is best to remove any other extraneous columns (such as gene name), keep only the accession number and data value columns.
   Save the spreadhseet as a tab-delimitted text (.txt) file.

2. Import normalized expression data using the Data Import Editor (not the Wizard) under File, Import Data, Use Editor. Select Add Files, then use the browser to select file(s), click 'Import All'

3. Once the data is imported, bring in the annotations by selecting Add Database Tables (radio button). A database connectivity window will open up, enter the following fields in the form:
   Database source: **Oracle** (default)
   Server Name: **turgidson**
   Port: **1521**
   Database Name: **expr**
   Owner: % (default)
   User Id: **data_integration**
   Password: **data_integration**

   Then click 'Connect'
   Click the tab labelled 'Advanced'
   Then copy in the SQL statement used to retrieve chip annotations, using the array layout name corresponding to your chip set. This can be found on the shared drive in the OmniViz folder.

ex: **select accession_number, gene_symbol, gene_name chromosome, cytogenetic_position, broad_biological_process_list, biological_process_list, broad_function_list, function_list, broad_cellular_component_list, cellular_component_list from chip_annotations where array_layout_name like 'Mouse: BMAP%'**

```
select accession_number, gene_symbol, gene_name
chromosome, cytogenetic_position, broad_biological_process_list,
biological_process_list, broad_function_list, function_list,
broad_cellular_component_list, cellular_component_list
from chip_annotations
where array_layout_name like 'Mouse: BMAP%'
```

4. Once the annotations have been imported, name the resulting data set, and then shift-click to highlight both data sets, and select 'Merge'.

Select 'Concatenate Side-to_side' and 'Combine columns with same headers'

Appendix II.C GenMAPP

GenMAPP (Gene MicroArray Pathway Profiler) is a software application designed to visualize gene expression data on maps representing biological pathways and groupings of genes.

Using the program consists of first creating an image file or MAPP which consists of a collection of gene objects and describes a biological function or event. Each gene object in the .mapp file is linked by its' accession number to a local database. The next step is to create a data file of the gene expression results, which will also be linked to the genes on the .mapp by accession number. Gene objects on the .mapp are then color- coded according to criteria established by the user, which can include log ratios, fold changes, and signficance statistics. Thus it is a very flexible program for displaying gene expression data according to user-defined characteristics.

Detailed Help files for the entire program are located at:
http://www.genmapp.org/GenMAPPHelp/HelpFiles/GenMAPP.htm

GenMAPP 1.0 Beta Protocol

1. Create a spreadhseet containing a list of genes representing the pathway or grouping of interest. The MAPP Builder tool will automatically generate the .mapp file from this spreadsheet. The spreadsheet must be in the following format:
   The first row must contain the column headings **Primary, PrimaryType, Label, Head, Remarks, and MAPPName**, spelled identically and in that order. Primary represents the gene ID, which is the (GenBank) accession number that identifies it on the slide. **PrimaryType** is **G** for GenBank. **Label** is the text you wish to identify the gene with on the MAPP, typically the gene symbol. **Head** is the title of the backpage file, containing information about the gene. **Remarks** is additional information that will appear in the backpage file for that gene. **MAPPName** is the filename for the MAPP file. The fields **Primary**, **PrimaryType**, and **MAPPName** must be present for each gene. If you wish to include a gene on the MAPP, but it is not present on the microarray slide, enter 'absent' in the Primary field, and the gene will show up, but will not be linked to experimental data.
   Save this file in a tab-delimited format (.txt) and then run the MAPP Builder program.

2. Open the MAPP file that has been created, and use the graphic tools in the toolbox to finish designing the graphical representation. Gene objects can be moved around, and arrows, text, and simple shapes can be added to generate a representation of a biological pathway or process. For examples, look at some of the MAPPs in the archive files.

3. Import expression data, using the Expression Dataset Manager (left-click 'Data' then 'Expression Dataset Manager). Fill in the top portion with the experiment name, and any remarks or notes. Begin the import process by clicking 'New' under 'Expression Data'

   The data file needs to be a comma-delimited *.csv* format. As the data is imported, the expression manager will check each accession number to determine if it exists in the local database. If the accession numbers are not found in the local database, it be added to an 'Exceptions' file. It is then necessary to process the exceptions file, which will add those accession numbers into the local database.

4. Create the color set file for displaying data. Criteria for assigning colors are based on the data columns imported in the expression data file, and can be set to whatever is desired. For each color, fill out the Criteria Builder section by choosing the column and value range for assigning the color, and fill in the 'label in legend' to describe the criteria. Multiple color sets can be generated and applied to any .mapp file.

5. After creating and saving the color set apply it to the .mapp image. (left-click 'Data' then 'Choose Expression Dataset') The result is the .mapp image with each of the gene objects color-coded to represent data values, and a legend describing the experiment and the criteria for each color.

Appendix III.A QChip HTML Query Form

This is the HTML code for the query form. The file is q1.html

```
<head>
<meta http-equiv="content-type" content="text/html;charset=iso-8859-1">
<meta name="generator" content="Adobe GoLive 5">
<title>QChip</title>
</head>
<body bgcolor="#ffffff">

 <h3>Retrieve information on microarray chip sequences</h3>

<p><label><h4>Select Array Layout</h4></label></p>

<form name="Input Form" action="http://bic0659.ohsu.edu/servlet/Servlet15" method="get">

<select name="Array_Layout" size="3" tabindex="1">
<option value="Human: R66 11-02-01" selected>Human: R66 11-02-01</option>
<option value="Mouse: NIA 12-18-01">Mouse: NIA 12-18-01</option>
<option value="Mouse: BMAP 3-29-02">Mouse: BMAP 3-29-02</option>
</select>

<label><h4>Search by:</h4></label>
<select name="SearchBy" size="4" multiple>
<option value="Accession Number">Accession Number</option>
<option value="Gene Symbol">Gene Symbol</option>
<option value="Gene Name" selected>Gene Name</option>
<option value="GO Term">GO Term</option>
</select>

<label><h4>Value</h4></label>
<input type="text" name="Q_string" size="48">
<input type="submit" name="submitButtonName">
</form>

 <p><b> Notes: </b></p>
<ul>
<li> Use % to replace unknown characters, for gene symbol query only <br>
     example:   TGFB% returns all gene symbols that begin with TGFB <BR><br>
<li> The Gene Name and GO Term searches will retrieve all sequences containing the        query string
<br><br>
<li> <a href = "http://www.godatabase.org/cgi-bin/go.cgi" target = "_BLANK"> Find GO terms with
AmiGO browser </a><br>

</body>
</html>
```

Appendix III.B QChip servlet source code

```
// QChip Query Interface source code
// Michelle Bobo
// 18 July 2002

import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.util.*;
import sqlj.runtime.*;
import sqlj.runtime.ref.*;
import java.sql.*;
import oracle.sqlj.runtime.Oracle;


public class Servlet15 extends HttpServlet {

  // Initialize global variables

  public void init(ServletConfig config) throws ServletException {
    super.init(config);
  }

// declare the results iterator class

#sql private static iterator ResultIteratorNamedClass (
  String accession_number,
  String gene_symbol,
  String gene_name,
  String biological_process_list,
  String function_list,
  String cellular_component_list,
  String position,
  String unigene_cluster_id
  );

    //Process the HTTP Get request

  public void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException
 {
    response.setContentType("text/html; charset=WINDOWS-1252");
    PrintWriter out = response.getWriter();
    out.println("<html><head><title> Search Results </title></head>");
    out.println ("<body>");
    out.println ("<H2> Search Results </H2>");

    String querystring = request.getQueryString();
    String layout = request.getParameter("Array_Layout");
    String search = request.getParameter("SearchBy");
    String query = request.getParameter("Q_string");
    String query2 = "%" + query + "%";
    String maketable = "<table width=100% border=1> <tr> <th>Position </th> <th>Unigene_ID </th>
<th>Accession Number </th> <th>Gene Symbol  </th> <th>Gene Name  </th> <th>Biological
Processes </th> <th>Molecular Functions  </th> <th>Cellular Localization  </th>";
```

```
    String organism;
    organism = make_organism (layout);
    String link = "http://bic0659.ohsu.edu/servlet/SpreadsheetFormat";

    // Return query parameters
    out.println ("<H4> Array layout: " + layout + "</H4>");
    out.println ("<H4> Search by: " + search + "</H4>");
    out.println ("<H4> Query string: " + query + "</H4>");
    out.println ("<H4> Organism: " + organism + "</H4>");
    out.println ("<A HREF=" + link + "?" + querystring + "> Return results in spreadsheet format </A>");


// Connect to database
    try {
    Oracle.connect(
      "jdbc:oracle:thin:@turgidson:1521:expr",
      "data_integration" , "data_integration"
      );

      ResultIteratorNamedClass results_iterator;

      //user searches for gene symbol
      if (search.equals("Gene Symbol"))
      {

        String upper_query;
        upper_query = query.toUpperCase();

        #sql results_iterator = {
          SELECT distinct position, unigene_cluster_id,
                 accession_number, gene_symbol, gene_name,
                 biological_process_list, function_list,
                 cellular_component_list
          FROM   chip_annotations
          WHERE  organism like :organism
               and array_layout_name like :layout
               and (gene_symbol like:query or gene_symbol_upper like :upper_query)
          };

          out.println(maketable); //creates HTML table
          while (results_iterator.next())
          {
            String url;
            url = make_link (results_iterator.unigene_cluster_id());
            out.println("<tr>");
            out.println("<td>" + results_iterator.position() + "  </td>");
            out.println("<td>" + "<A HREF = \"" + url + "\" + TARGET = \"_BLANK\">" +
results_iterator.unigene_cluster_id() + "</A> </td>");
            out.println("<td>" + results_iterator.accession_number() + "  </td>");
            out.println("<td>" + results_iterator.gene_symbol() + " </td>");
            out.println("<td>" + results_iterator.gene_name() + " </td>");
            out.println("<td>" + results_iterator.biological_process_list() + " </td>");
            out.println("<td>" + results_iterator.function_list() + " </td>");
            out.println("<td>" + results_iterator.cellular_component_list() + " </td>");
            } //end of while loop
            out.println ("</table>");
        results_iterator.close();
      } //end if

    else if (search.equals("GO Term"))
    {
     #sql results_iterator = {
      SELECT  position, unigene_cluster_id,
              accession_number, gene_symbol, gene_name,
              biological_process_list, function_list,
              cellular_component_list
      FROM   chip_annotations
      WHERE  (biological_process_list like :query2 or function_list like :query2
              or cellular_component_list like :query2)
             and array_layout_name like :layout
      };

      out.println(maketable);   //creates HTML table
      while (results_iterator.next())
      {
       String url;
       url = make_link (results_iterator.unigene_cluster_id());
       out.println("<tr>");
       out.println("<td>" + results_iterator.position() + "</td>");
       out.println("<td>" + "<A HREF = \"" + url + "\" + TARGET = \"_BLANK\">" +
results_iterator.unigene_cluster_id() + "</A> </td>");
       out.println("<td>" + results_iterator.accession_number() + "</td>");
       out.println("<td>" + results_iterator.gene_symbol() + "</td>");
       out.println("<td>" + results_iterator.gene_name() + "</td>");
       out.println("<td>" + results_iterator.biological_process_list() + "</td>");
       out.println("<td>" + results_iterator.function_list() + "</td>");
       out.println("<td>" + results_iterator.cellular_component_list() + "</td>");
       } //end of while loop
       out.println ("</table>");
       results_iterator.close();
     } //end if

     else if (search.equals("Gene Name"))
    {
     #sql results_iterator = {
      SELECT  position, unigene_cluster_id,
              accession_number, gene_symbol, gene_name,
              biological_process_list, function_list,
              cellular_component_list
      FROM   chip_annotations
      WHERE  gene_name like :query2
             and array_layout_name like :layout
      };

     out.println(maketable);  //creates HTML table
     while (results_iterator.next())
      {
       String url;
       url = make_link (results_iterator.unigene_cluster_id());
       out.println("<tr>");
       out.println("<td>" + results_iterator.position() + "</td>");
```

```
        out.println("<td>" + "<A HREF = \"" + url + "\" + TARGET = \"_BLANK\">" +
results_iterator.unigene_cluster_id() + "</A> </td>");
        out.println("<td>" + results_iterator.accession_number() + "</td>");
        out.println("<td>" + results_iterator.gene_symbol() + "</td>");
        out.println("<td>" + results_iterator.gene_name() + "</td>");
        out.println("<td>" + results_iterator.biological_process_list() + "</td>");
        out.println("<td>" + results_iterator.function_list() + "</td>");
        out.println("<td>" + results_iterator.cellular_component_list() + "</td>");
        } //end of while loop
        out.println ("</table>");
        results_iterator.close();
      } //end if

     else if (search.equals("Accession Number"))
     {
      #sql results_iterator = {
      SELECT a.unigene_cluster_id, a.accession_number, a.gene_symbol, a.gene_name,
            a.biological_process_list, a.function_list,
            a.cellular_component_list, a.position
      FROM chip_annotations a,
         gene_accession_numbers n
      WHERE  n.unigene_cluster_id = a.unigene_cluster_id
      AND array_layout_name like :layout
      AND n.accession_number like :query
      };

      out.println(maketable);  //creates HTML table
      while (results_iterator.next())
       {
        String url;
        url = make_link (results_iterator.unigene_cluster_id());
        out.println("<tr>");
        out.println("<td>" + results_iterator.position() + "</td>");
        out.println("<td>" + "<A HREF = \"" + url + "\" + TARGET = \"_BLANK\">" +
results_iterator.unigene_cluster_id() + "</A> </td>");
        out.println("<td>" + results_iterator.accession_number() + "</td>");
        out.println("<td>" + results_iterator.gene_symbol() + "</td>");
        out.println("<td>" + results_iterator.gene_name() + "</td>");
        out.println("<td>" + results_iterator.biological_process_list() + "</td>");
        out.println("<td>" + results_iterator.function_list() + "</td>");
        out.println("<td>" + results_iterator.cellular_component_list() + "</td>");
        } //end of while loop
        out.println ("</table>");
        results_iterator.close();
      } //end if

     else {out.println("please select a different option");}

      } catch (SQLException e) {out.println("SQLException " + e);}

      try {Oracle.close();}
      catch (SQLException e) {out.println("SQLException " + e);}

      out.println ("</body></html>");

  }// end doGET()
```

```
// This method creates the URL for the link to UniGene
private String make_link (String uci)
{
 String org = uci.substring(0,2);
 String id_num = uci.substring(3);
 String url = "http://www.ncbi.nlm.nih.gov/UniGene/clust.cgi?ORG=" + org + "&CID=" + id_num;
 return url;
 } // end make_link

 // This method determines the organism from the array layout
 private String make_organism (String layout)
 {
  String species = null;
  if (layout.equals("Human: R66 11-02-01"))
    {species = "Homo sapiens";}
  else if (layout.equals("Mouse: BMAP 3-29-02"))
    {species = "Mus musculus";}
  else if (layout.equals("Mouse: NIA 12-18-01"))
    {species = "Mus musculus";}
  return species;
  } // end make_organism

} // end Servlet
```

Appendix III.C Spreadsheet Format Servlet Source Code

```
/* This is the servlet that returns the QChip query results in
spreadsheet format
Michelle Bobo
18 July 2002
*/

import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
import java.util.*;
import sqlj.runtime.*;
import sqlj.runtime.ref.*;
import java.sql.*;
import oracle.sqlj.runtime.Oracle;


public class SpreadsheetFormat extends HttpServlet {

  // Initialize global variables

  public void init(ServletConfig config) throws ServletException {
    super.init(config);
  }

// declare the iterator class

#sql private static iterator ResultIteratorNamedClass (
  String accession_number,
  String gene_symbol,
  String gene_name,
  String biological_process_list,
  String function_list,
  String cellular_component_list,
  String position,
  String unigene_cluster_id
  );

   //Process the HTTP Get request

  public void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
    response.setContentType("text/html; charset=WINDOWS-1252");
    PrintWriter out = response.getWriter();
    out.println("<html><head><title> Search Results </title></head>");
    out.println ("<body>");
    out.println ("<H2> Search Results </H2>");


    String layout = request.getParameter("Array_Layout");
    String search = request.getParameter("SearchBy");
    String query = request.getParameter("Q_string");
    String query2 = "%" + query + "%";
    String maketable = "Position;Unigene_ID;Accession Number;Gene Symbol;Gene Name;Biological
Processes;Molecular Functions;Cellular Localization; ";
```

```
  String organism;
  organism = make_organism (layout);

  out.println ("<H4> Array layout: " + layout + "</H4>");
  out.println ("<H4> Search by: " + search + "</H4>");
  out.println ("<H4> Query string: " + query + "</H4>");
  out.println ("<H4> Organism: " + organism + "</H4>");

// Connect to database
  try {
  Oracle.connect(
    "jdbc:oracle:thin:@turgidson:1521:expr",
    "data_integration" , "data_integration"
    );

    ResultIteratorNamedClass results_iterator;

//user searches for gene symbol
    if (search.equals("Gene Symbol"))
    {
     String upper_query;
     upper_query = query.toUpperCase();

     #sql results_iterator = {
      SELECT position, unigene_cluster_id,
             accession_number, gene_symbol, gene_name,
             biological_process_list, function_list,
             cellular_component_list
      FROM   chip_annotations
      WHERE  organism like :organism
            and array_layout_name like :layout
            and gene_symbol_upper like :upper_query
      };

      out.println("<PRE>" + maketable + "</PRE>"); //prints column headers
      while (results_iterator.next())
      {
       out.print("<PRE>");
       out.println(results_iterator.position() + ";" + results_iterator.unigene_cluster_id() + ";"  +
results_iterator.accession_number() + ";" + results_iterator.gene_symbol() + ";" +
results_iterator.gene_name() + ";"  + results_iterator.biological_process_list() + ";" +
results_iterator.function_list() + ";"  + results_iterator.cellular_component_list() + ";");
       }
       results_iterator.close();
       out.println ("</PRE>");
     } //end if

    else if (search.equals("GO Term"))
    {
     #sql results_iterator = {
      SELECT  position, unigene_cluster_id,
             accession_number, gene_symbol, gene_name,
             biological_process_list, function_list,
             cellular_component_list
      FROM   chip_annotations
      WHERE  (biological_process_list like :query2 or function_list like :query2
            or cellular_component_list like :query2)
           and array_layout_name like :layout
       };

       out.println(maketable);   //prints column headers
       while (results_iterator.next())
       {
         out.print("<PRE>");
         out.println(results_iterator.position() + ";" + results_iterator.unigene_cluster_id() + ";" +
results_iterator.accession_number() + ";" + results_iterator.gene_symbol() + ";" +
results_iterator.gene_name() + ";"  + results_iterator.biological_process_list() + ";" +
results_iterator.function_list() + ";" + results_iterator.cellular_component_list() + ";");
          }
         results_iterator.close();
         out.println ("</PRE>");
      } //end if

      else if (search.equals("Gene Name"))
     {
      #sql results_iterator = {
        SELECT  position, unigene_cluster_id,
               accession_number, gene_symbol, gene_name,
               biological_process_list, function_list,
               cellular_component_list
        FROM   chip_annotations
        WHERE  gene_name like :query2
              and array_layout_name like :layout
        };

      out.println(maketable);  //prints column headers
      while (results_iterator.next())
       {
         out.print("<PRE>");
         out.println(results_iterator.position() + ";" + results_iterator.unigene_cluster_id() + ";"  +
results_iterator.accession_number() + ";" + results_iterator.gene_symbol() + ";" +
results_iterator.gene_name() + ";"  + results_iterator.biological_process_list() + ";" +
results_iterator.function_list() + ";"  + results_iterator.cellular_component_list() + ";");
          }
         results_iterator.close();
         out.println ("</PRE>");
      } //end if

     else if (search.equals("Accession Number"))
     {
      #sql results_iterator = {
      SELECT a.unigene_cluster_id, a.accession_number, a.gene_symbol, a.gene_name,
             a.biological_process_list, a.function_list,
             a.cellular_component_list, a.position
      FROM chip_annotations a,
         gene_accession_numbers n
      WHERE  n.unigene_cluster_id = a.unigene_cluster_id
      AND array_layout_name like :layout
      AND n.accession_number like :query
      };

      out.println(maketable);  //prints column headers
```

```
        while (results_iterator.next())
         {
          out.print("<PRE>");
          out.println(results_iterator.position() + ";" + results_iterator.unigene_cluster_id() + ";"   +
results_iterator.accession_number() + ";" + results_iterator.gene_symbol() + ";" +
results_iterator.gene_name() + ";"   + results_iterator.biological_process_list() + ";" +
results_iterator.function_list() + ";"   + results_iterator.cellular_component_list() + ";");
          }
          results_iterator.close();
          out.println ("</PRE>");
        } //end if

       else {out.println("please select a different option");}

        } catch (SQLException e) {out.println("SQLException " + e);}

        try {Oracle.close();}
        catch (SQLException e) {out.println("SQLException " + e);}

        out.println ("</body></html>");

  }// end doGET()

  private String make_organism (String layout)
  {
   String species = null;
   if (layout.equals("Human: R66 11-02-01"))
     {species = "Homo sapiens";}
   else if (layout.equals("Mouse: BMAP 3-29-02"))
     {species = "Mus musculus";}
   else if (layout.equals("Mouse: NIA 12-18-01"))
     {species = "Mus musculus";}
   return species;
   } // end make_organism

}// end SpreadsheetFormat
```

Appendix III.D Additional QChip documentation

Updating the program

The QChip interface has been designed to be easy to update in order to accommodate additional slide sets. The following steps will update the program when a new slide set becomes available.

1. Update the chip_annotations view in the data warehouse. (See Appendix II.A), adding the new array layout name to the final clause in the select statement. This will make annotations for that slide set available to the QChip program.

2. Update the HTML query form (Appendix III.A) , adding the new array layout to the "Array Layout" select list.

3. Update the make_organism method at the end of the servlet code. Add a new 'else if' clause stating the organism for the new slide set.

The program passes the layout name exactly to the SQL query statement, so as long as the layout name is accurate in both the chip_annotations view and the query form, the update should be very easy.

References

1. Lockhart, D.J. and E.A. Winzeler, *Genomics, gene expression and DNA arrays.* Nature, 2000. **405**(6788): p. 827-36.
2. International Conference on Systems BIology, 2001, http:www.icsb2001.org/what_is.html
3. Schena, M., et al., *Parallel human genome analysis: Microarray-based expression monitoring of 1000 genes.* Proceedings of the National Academy of Sciences, 1996. **93**: p. 10614-10619.
4. Brazma, A., et al., *Minimum information about a microarray experiment (MIAME)-toward standards for microarray data.* Nat Genet, 2001. **29**(4): p. 365-71.
5. Gene Views, Center for Biomarker Discovery, 2002, http://medir.ohsu.edu/~geneview/
6. Tufte, E., *The Visual Display of Quantitative Information.* 2nd ed. 2001: Graphics Press.
7. Lander, E.S., et al., *Initial sequencing and analysis of the human genome.* Nature, 2001. **409**(6822): p. 860-921.
8. Venter, J.C., et al., *The sequence of the human genome.* Science, 2001. **291**(5507): p. 1304-51.
9. Hubbard, T., et al., *The Ensembl genome database project.* Nucleic Acids Res, 2002. **30**(1): p. 38-41.
10. Stein, L., *Genome annotation: from sequence to biology.* Nat Rev Genet, 2001. **2**(7): p. 493-503.
11. Hill, D.P., et al., *Program description: Strategies for biological annotation of mammalian systems: implementing gene ontologies in mouse genome informatics.* Genomics, 2001. **74**(1): p. 121-8.
12. Nadeau, J.H., et al., *Sequence interpretation. Functional annotation of mouse genome sequences.* Science, 2001. **291**(5507): p. 1251-5.
13. Beckers, J. and M. Hrabe de Angelis, *Large-scale mutational analysis for the annotation of the mouse genome.* Curr Opin Chem Biol, 2002. **6**(1): p. 17-23.
14. Kawai, J., et al., *Functional annotation of a full-length mouse cDNA collection.* Nature, 2001. **409**(6821): p. 685-90.
15. Consortium, T.G.O., *Gene Ontology: tool for the unification of biology.* Nature Genetics, 2000. **25**: p. 25-29.
16. Karp, P.D., *An ontology for biological function based on molecular interactions.* Bioinformatics, 2000. **16**(3): p. 269-85.
17. *Creating the gene ontology resource: design and implementation.* Genome Res, 2001. **11**(8): p. 1425-33.
18. Stead, W.W., et al., *Integration and beyond: linking information from disparate sources and into workflow.* J Am Med Inform Assoc, 2000. **7**(2): p. 135-45.
19. Stanford, *Stanford Online Universal Resource for Clones and EST's (SOURCE)* 2001.

20. University, T.J., *CloneUpdaterURL* 2001.
21. Wayne State University, I.S.a.B.D., *Onto-ExpressURL* 2001.
22. Array Organizing Tool (Arrogant), http://lethargy.swmed.edu/index.asp
23. Mouse Gene Associations, 2002, ftp://ftp.informatics.jax.org/pub/informatics/reports/gene_association.mgi
24. Quackenbush, J., *Viva la revolution! A report from the FANTOM meeting.* Nat Genet, 2000. **26**(3): p. 255-6.
25. Hall, M., *Core Servlets and Java Server Pages.* 2000, Upper Saddle River, NJ: Prentice Hall.
26. Price, J., *Java Programming with Oracle SQLJ.* 2001, Sebastopol, CA: O'Reilly & Associates, Inc.
27. Welcsh, P.L., et al., *BRCA1 transcriptionally regulates genes involved in breast tumorigenesis.* Proc Natl Acad Sci U S A, 2002. **99**(11): p. 7560-5.
28. Liu, T., et al., *Developing a strategy to define the effects of insulin-like growth factor-1 on gene expression profile in cardiomyocytes.* Circ Res, 2001. **88**(12): p. 1231-8.
29. De La Fuente, C., et al., *Gene expression profile of HIV-1 Tat expressing cells: a close interplay between proliferative and differentiation signals.* BMC Biochem, 2002. **3**(1): p. 14.
30. Cadet, J.L., et al., *Temporal profiling of methamphetamine-induced changes in gene expression in the mouse brain: evidence from cDNA array.* Synapse, 2001. **41**(1): p. 40-8.
31. Jayanthi, S., et al., *Methamphetamine causes coordinate regulation of Src, Cas, Crk, and the Jun N-terminal kinase-Jun pathway.* Mol Pharmacol, 2002. **61**(5): p. 1124-31.
32. Wolf, D., C.P. Gray, and A. de Saizieu, *Visualising gene expression in its metabolic context.* Brief Bioinform, 2000. **1**(3): p. 297-304.
33. Onto-Express, Wayne State University, Intelligent Systems and Bioinformatics Department, 2001, http://vortex.cs.wayne.edu/Projects.html
34. Dowell, R.D., et al., *The Distributed Annotation System.* BMC Bioinformatics, 2001. **2**(1): p. 7.
35. Biodas Project, 2002, http://biodas.org
36. Kanehisa, M., et al., *The KEGG databases at GenomeNet.* Nucleic Acids Res, 2002. **30**(1): p. 42-6.