# Animal Models of Diabetic Complications Consortium
# (U01 DK60966-01)

## Annual Report
## (2005-2006)

## "Coordinating and Bioinformatics Unit of the AMDCC"
## Medical College of Georgia
## Augusta, Georgia

## Richard A. McIndoe, Ph.D.
## Program Director

**Richard A. McIndoe, Ph.D.**
**Medical College of Georgia**
**Center of Biotechnology and Genomic Medicine**
**Augusta, GA 30912**
**Phone: (706) 721-3542 Fax: (706) 721-3688**
**Email: rmcindoe@mail.mcg.edu**

# Table of Contents


**Page**

**Part A: Principal Investigator's Summary**

**1. Project Accomplishments** **4**

# Animal Models of Diabetic Complications Consortium
# (U01 DK60966-01)

## Part A:

## Principal Investigator's Summary

## 1. Program Accomplishments:

## Brief Overview of System

The Coordinating and Bioinformatics unit is responsible for the creation of the informatics infrastructure of the consortium as well as facilitating the efforts of the mouse engineering centers. Our programming paradigm is to develop software systems based on an n-tier architecture, where we create the presentation layer, business logic and data layer into separate software systems. As a reminder, figure 1 presents the system we have created for the consortium. These systems have been developed to minimize maintenance, but provide a robust scalable model for future growth and interactions at the national level with other organism databases.

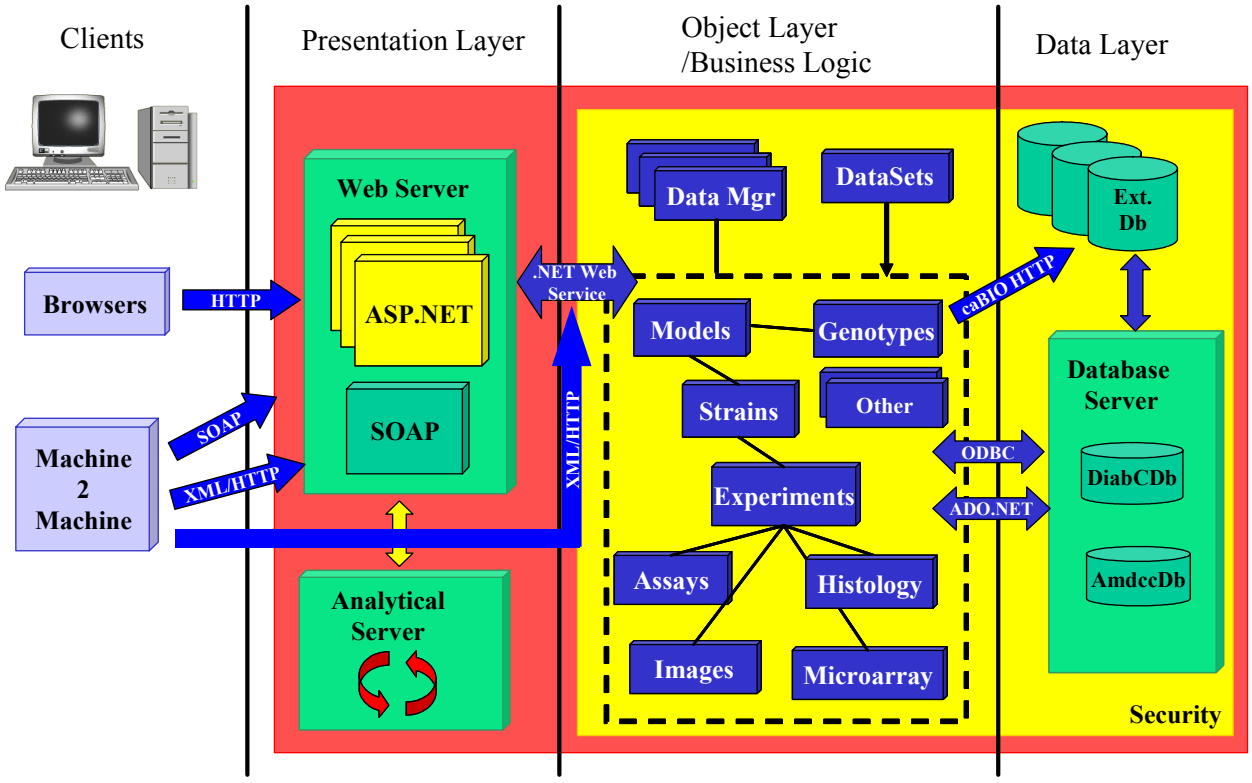



Figure 1. AMDCC n-tier design architecture

We designed the system using the unified modeling language (UML) and Powerdesigner (SyBase). Additional details regarding the requirements analysis, controlled vocabulary and creation of the AMDCC software systems including AMDCC object/data model design and implementation on the web portal itself http://www.amdcc.org/bioinformatics/bioinformatics.aspx.

**AMDCC Data Entities** Because the AMDCC generates both quantitative and qualitative data, one of the technical challenges of creating the AMDCC web portal was to accommodate the diversity of data and experimental designs. To provide the scientific community with a comprehensive system that is contextual and flexible, we developed a software architecture that unifies the various experimental entities.

**Experiments** All data entered into the system is done in the context of an "Experiment". The Experiment domain is quite generic and provides a unifying structure to integrate the heterogeneous data types collected by the members of the consortium. This domain allows users to specify animals, protocols, assays, gene expression data, histology, images and independent variables used in an experiment. Of note is the ability to define the independent variables (termed Experimental Conditions in the portal) which allows investigators to define their results in precise terms, such as taking samples at timed intervals or varying dosage of drugs. This aspect provides clear ways to compare data across different experiments.

**Models and Strains** Investigators use the words Model and Strain interchangeably. However, we separated the two concepts with respect to the IT structure of the AMDCC web portal since a particular mouse strain could be a model for more then one complication. Animal models are defined by the complication type, diabetes type, species and strain and can accommodate diabetes and complication induction protocols. Strains are defined by the genetic manipulation and breeding strategy. As of this writing, the system holds data on 76 models (under development) and 55 AMDCC-created strains (51 mouse and 4 pig strains).

**Protocols and Publications** One of the products of the consortium are the protocols developed by the membership. The various diabetic complication subcommittees have developed standard protocols for many of the phenotype assays and procedures performed on the animal models. Protocols can be attached to virtually any entity in the system and are conceptually grouped (eg. array manufacturing, diabetes induction, staining, breeding, etc.).

All the publications generated by consortium members must go through the AMDCC Publication Subcommittee before submission to a scientific journal. All publications that are *in press* or published are available to the general public. Due to copyright considerations, we provide only the publication citation and a hyperlink to the PubMed reference for published manuscripts.

**AMDCC Data Analysis** A number of analytical tools have been incorporated into the web portal to facilitate data analysis. The basic statistics interface provides an easy and flexible way to calculate basic statistical information. Users select the assays and strains they want to analyze as well as any filter criteria. The system will automatically calculate the mean, standard deviation, median, min, max, variance, skewness and kurtosis for each assay/strain combination. If the query contains both male and female data, the system will automatically do these calculations for the combined sexes as well as males and females separately.

The T Test interface provides a simple way for users to perform an unpaired T Test to assess whether two groups of animals are significantly different from each other. Users select the assay and then strain and filter criteria for each group to be tested and a Student's T value, p value, upper/lower confidence limits and group means, standard deviations and number are all calculated.

The ANOVA analysis interface supports both automated One Way and Two Way ANOVA analyses for both balanced and unbalanced experimental designs. Once the filter criteria are selected, the user is presented with the list of assays to analyze and two analytical

options, which automates what strain comparisons they want to make once the ANOVA is completed. The Two Way ANOVA currently only assesses Strain by Sex interactions. In addition, we use Tukey's HSD when calculating p values for all pairwise comparisons in all ANOVA analyses.

The correlations interface allows users to examine the relationship between two assays using Pearson's correlation coefficient, R squared, covariance, and plots the data with the regression line.

**AMDCC Data Visualization** In addition to analytical tools, users can generate charts on-the-fly based on any filter criteria. Data from both Experiments and ad-hoc datasets can be plotted. All charts provide error bars when appropriate and a fully editable interface to change the appearance, including titles and axis labels. Additionally, we use tooltips to provide details of the data by moving the mouse pointer over the data point or column. The system can generate column charts, line charts and XY plots.

**AMDCC Data Search and Retrieval** One of the key aspects of the web portal is the ability to search and download the data generated by the AMDCC membership. The data entities covered by the individual search pages include Animals, Assays, Complications, Experiments, Histology, Investigators, Microarrays, Models, Publications, Protocols and Strains.

Data from individual experiments can be downloaded as an Excel spreadsheet by simply navigating to the Experiment Definition page for the Experiment and clicking on the Browse Data link at the top of the page. The requested data is then displayed in a data grid that can be explored and downloaded immediately.

To simultaneously search and download data from multiple experiments, users can explore the data using the Create DataSet page. A dataset is a generic class used to construct complicated data queries across all the data generated by consortium, thus providing a mechanism for higher order analysis (e.g. meta analysis). Investigators can build sophisticated queries using all the objects in the AMDCC object model. For example, one can build a query to extract all the assays performed on C57BL/6 mice with blood glucose measurements > 200 mg/dl for at least 4 weeks post diabetes onset and induced with STZ 20 mg/kg body weight and fed on a high fat diet starting at 4 weeks of age for 3 months. *Ad-hoc* datasets can be downloaded as an Excel spreadsheet or used by the analysis and visualization tools at the web portal.

# AMDCC Web Portal Updates (2005-2006)

During this last year, the AMDCC website has undergone many changes. Some of these changes have been technological advancements for the website as a whole while others have been more functional additions and enhancements. The following sections will describe these changes in more detail with example figures presented when appropriate.

# Technological Advances

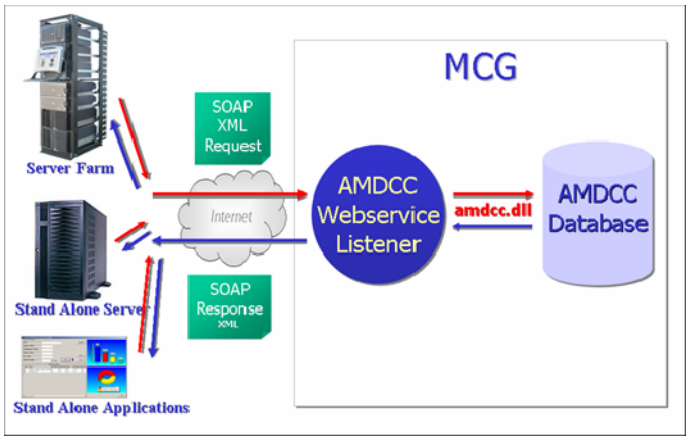
Based on a number of conversations with the membership, we have worked on some software components that provide increased flexibility and data sharing capabilities.

**AMDCC Web Services** One of the goals of the AMDCC is to share the data generated by the consortium with the scientific community. This would include both individual investigators as well as national data repositories such as the NCBI. Currently, individual investigators can download all the data generated by the AMDCC via the web portal. However, this strategy is not optimal for data repositories or other web sites that would like to retrieve our data. A better approach for these entities is to provide a web service that can be used for machine-to-machine transfer of data.

The term Web services describes a standardized way of integrating Web-based applications using XML, SOAP, WSDL and UDDI open standards over an Internet protocol backbone. XML provides structure to the data, SOAP is used to transfer the data, WSDL is used for describing the services available and UDDI is used for listing what services are available. Web services allow different applications from different sources to communicate with each other without time-consuming custom coding, and because all communication is in XML, Web services are not tied to any one operating system or programming language. For example, Java can talk with Perl, Windows applications can talk with UNIX applications.

Over the last year, we have completed the AMDCC Web Services component and have activated these services for the general public. Specifically, we created 12 web services that provide access to the various data domains stored and capture by the consortium members. All external SOAP connections assume the user is the general public and as such are limited to data that has been released to the public. Each of the web services focuses on a specific type of data. For example, the ExperimentWebService connection will retrieve data related to the AMDCC experiments stored in the system. As illustrated in Figure 2, a connection via the web services can be made by a variety of software systems. These can be a stand alone application that uses the web service to collect and display data or a stand alone server that runs nightly scripts to download the most recent data released to the public. Figure 2B provides a list of the available services with information on the data being retrieved using that web service. In addition to deploying the web services, we have also created help files, a tutorial and an example application that uses these services. All of these files and examples can be downloaded from the Bioinformatics subsection at the website.

**A**




**B**

| WebService | Comments |
|---|---|
| StrainWebService | AMDCC Strain Data |
| ProtocolWebService | AMDCC Protocol Data |
| ExperimentWebService | AMDCC Experiment Data |
| AnimalWebService | AMDCC Animal Data |
| BioAssayDataWebService | AMDCC Microarray Data |
| GenomeWebService | AMDCC Genomic Data |
| ExpFactorWebService | AMDCC Experiment Factor Data |
| PublicationWebService | AMDCC Publication Data |
| ModelWebService | AMDCC Model Data |
| SampleWebService | AMDCC Microarray Sample Data |
| BioAssayWebService | AMDCC Microarray Information |
| ExternalRefWebService | External References Maintained by AMDCC |

**Figure 2**. AMDCC webservices. **A**. Connections to the AMDCC web services can be made via a SOAP connection. **B**. The web services deployed on the AMDCC web portal.

**Analysis Engine Framework.** Last year we completed the object model and user interfaces necessary to allow consortium members to upload their microarray data to the system. This year we are focused on providing tools for the members and public to analyze and visualize the microarray data. During our requirements analysis, it became obvious that most of the typical analytical routines that users may want to perform require extended periods of time. For example, a simple hierarchical cluster of 5000 genes from 15 arrays can take several minutes to perform. Given we are using a web based user interface, it is unrealistic to keep a connection between the client (browser) and the servers during the analysis time. Therefore, we developed an Analysis Engine Framework that provides the infrastructure to queue and run lengthy analysis subroutines or algorithms asynchronously using our internal compute clusters. Example analysis routines could be various clustering algorithms or multivariate/classification analyses. The framework we created allows us to create any number of analytical routines and dynamically add them to the Analysis Engine without require us to re-compile the entire framework. Figure 3 illustrates the interaction between the various components of the framework. An analysis algorithm can be implemented by creating a class that has a Run method to start the analysis. The input and output of the analysis Run method must be serializable objects. An analysis Windows Service runs on each node of the Compute cluster. When an analysis needs to be run, the Windows Service on the compute node will load the assembly that implements the analysis (e.g.ExampleCalcs.dll), construct an instance of the type that implements the analysis (e.g. ExampleCalcs.ExampleCalcsImpl), runs the analysis by calling its Run method, and then unloads the assembly when the routine has completed. Analysis Request objects are stored in the Analysis Database along with the Analysis Type object that identifies the assembly to load (e.g. "ExampleCalcs") and the full name of the class that implements the analysis (e.g. "ExampleCalcs.ExampleCalcsImpl").

Stored Analysis Request objects are accessible through an Analysis Request Web Service. This service is used by all clients that wish to Queue and Show Results of an Analysis Request (e.g. www.amdcc.org). Using a web service gives the clients object oriented access to the Analysis Database without having to supply each with the latest object model assembly for the database. The Windows Services located on the compute nodes polls the Analysis Database via the Analysis Request Web

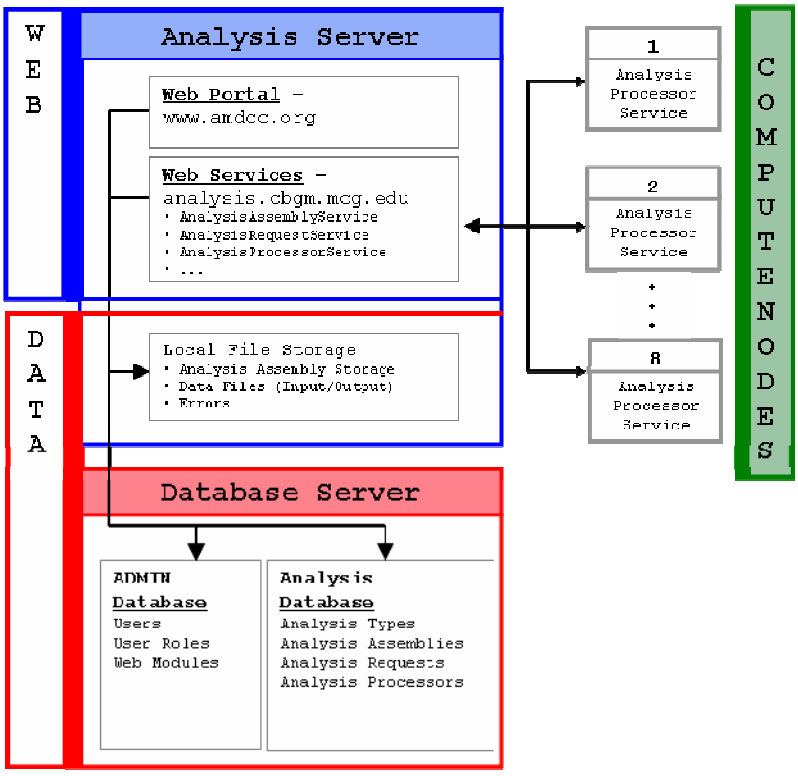



**Figure 3.** Diagram of Analysis Engine Framework. The framework is made of web services, a windows service and a database to hold the information on the type of analyses being requested as well as the software necessary to run the anlaysis.

Service at timed intervals to determine if any analyses need to be run. Each compute node can process up to 10 separate analysis threads, meaning our 8 compute nodes can handle at most 80 simultaneous analysis requests. We have implemented a round-robin technique for the compute nodes where nodes currently running one or more threads or have CPU intensive jobs running, will have a lower priority status to start a new job (thread) then those nodes not currently being used. Meaning, if the compute node is running a job and another node is idle, the job will run on the idle compute node. Once the analysis is completed, the output object becomes available to the client by accessing the Analysis Request Service.

The Analysis Engine Framework we created is extremely flexible and provides a way for us to share analytical routines as well as add new routines on-the-fly without requiring any updates to the framework or servers. Because of this, we are now in the position to add multiple clustering algorithms as well as classification systems for microarray data to the web portal. The first clustering algorithms we will implement are hierarchical clustering and Kmeans clustering.

**Improved Error Subsystem.** To improve our response to errors and problems that occur on the AMDCC web portal, we developed an error reporting subsystem to report unhandled exceptions that occur on the web site. Our programming paradigm is to anticipate and trap for errors before pages are posted to the server for execution. For example, the page may require the user to select an animal strain before continuing to post the page to the server. In this case, we have java script on the client that sends an error message to the page warning the user they can not continue until the strain is selected. However, there will be errors or bugs in the code that we did not anticipate. In these instances, the error will show up on an error page but unless the user notifies us of the error, we will not know that an error occurred and will not be able to fix the problem. To solve this issue, we created an error reporting subsystem that fires whenever an unhandled error occurs. The subsystem will create an email message with information on the user, browser, IP address and Host name, URL and raw URL during page request, the error message and the stack trace. Figure 4 is an example email with the text that is created by the subsystem. Once we receive the email, we can begin to investigate the problem and if the user information is available we can call the user and ask how to reproduce the error. Our response times are usually within the hour.

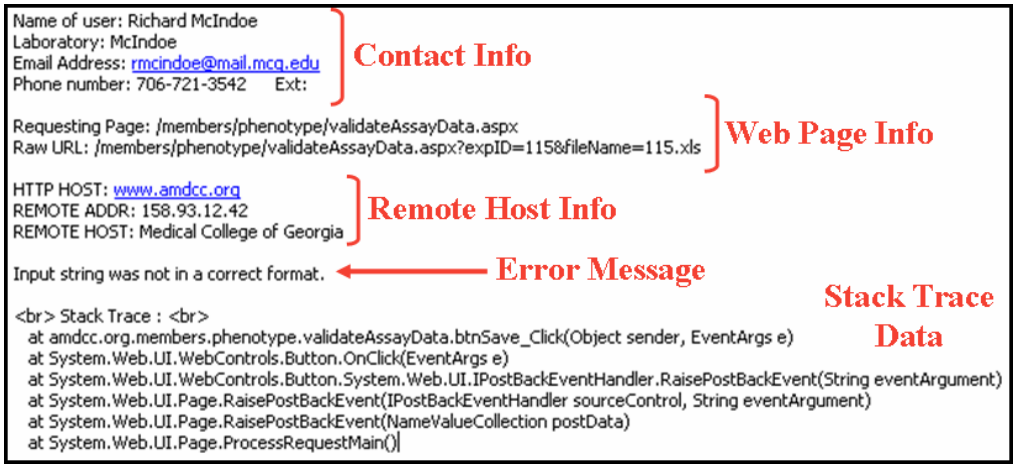



**Figure 4**. Example email sent by the AMDCC error subsystem.

# Web Portal Enhancements

Over the last year we have added a number of enhancements and improvements to the website with respect to the user interface, analytical methods and phenotype exploration.

**On-the-fly charting options for all charting functions.** During the last year, we completed augmenting the user interface for the various charting features of the web portal. Previously, the user could only customize the axis labels and chart titles text. Our goal was to allow the end user to have complete control over the look of the resulting chart. To accomplish this goal, we created a flexible menu structure that allows the user to choose or alter a number of visual features such as background colors and font text attributes. However, because the amount of screen space available is limited, we use a server control that can be activated by a right mouse click over the chart itself. Figure 5 presents the menu structure we created for the charting feature of the web portal. We allow the user to change the border color, error bar color, grid color, series colors, chart size as well as the font attributes (color, bold, italics, size and style) for all the axes and title text. The user can choose from 238 different colors. We provide both the color name and a small swatch to show what the color looks like.

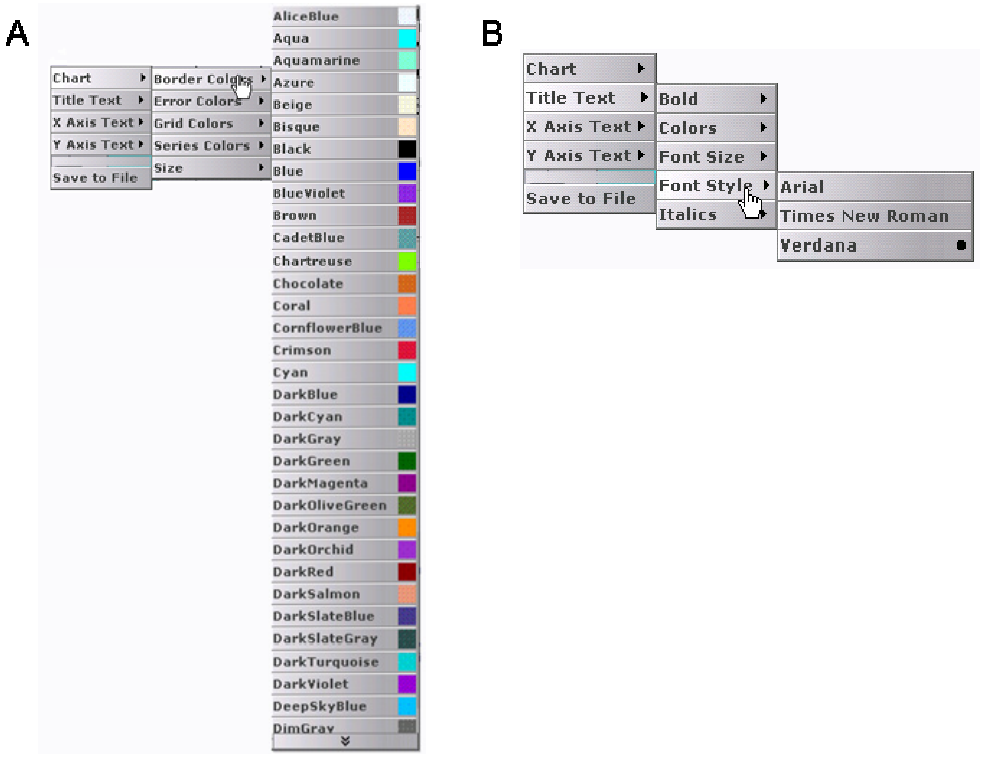



**Figure 5.** Flexible charting menu to alter the look and feel of the charts generated by the web portal. **A.** Options for changing the colors of the various chart features. The portal provides access to 238 different colors. **B**. Font attricbutes that can be changed for both axis labels and title text.

**Ease of movement between analysis tools and charting tools.** Although we provide both statistics and charting options for the end users, we did not have a method for moving quickly between the two exploration methods. For example, a user who has charted a number of strains for a particular assay may want to know if any of the strain combinations are statistically different from each other. To do this, they would have to go through the Statistical Exploration feature and re-select the experiment and all the filtering options. We have streamlined this aspect of the portal by providing a one-click option to move from a chart to statistics or from the statistics pages to a chart. For example, in Figure 6A we now provide two buttons that allow the user to perform either basic statistics or an ANOVA analysis for the data plotted in the chart. Figure 6B shows the one-click option for plotting either strain combinations or any number of strains based on the experiment and filter options used to do the analysis.

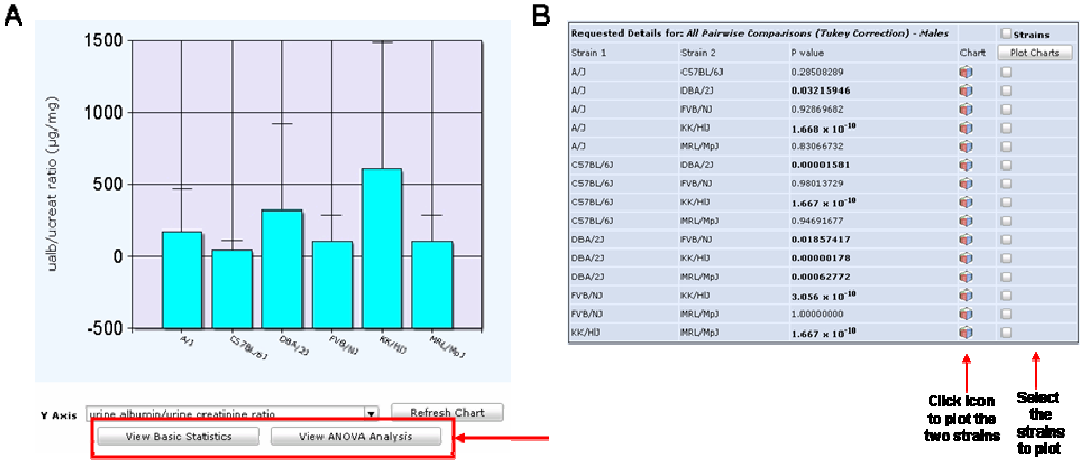



**Figure 6.** Ways to move between charts and statistics or statistics to charts. **A.** Two new buttons (outlined in the red box) have been included on the charting page for the user to view either the basic statistics or perform an ANOVA analysis on the data currently being charted. **B.** Charting options are now provided in both the basic statistics and ANOVA analysis pages. Here is an example from the ANOVA page. The user can either click the icon to see the two strains plotted or select any combination of strains to plot.

**Improved Data Retrieval for large Data Lists.** In order to improve the user's experience, we have re-designed a number of the data retrieval aspects for key drop down menus that provide access to large lists of items. For example, if we want to present a drop down list of all the mouse strains in the system, it would take some time to generate the HTML code and require the user to scroll through a very long list of strains to pick one. To simplify and enhance the experience, we have developed a solution based on AJAX (Asynchronous Java Execution) where the user can retrieve the data in real time by simply typing the beginning letters of the item and the data is retrieved from the server asynchronously and returned to the client. Figure 7 illustrates the use of the AJAX enabled control to retrieve a list of genes from the server rather then trying to pre-populate the drop down box with the thousands of genes and forcing the user

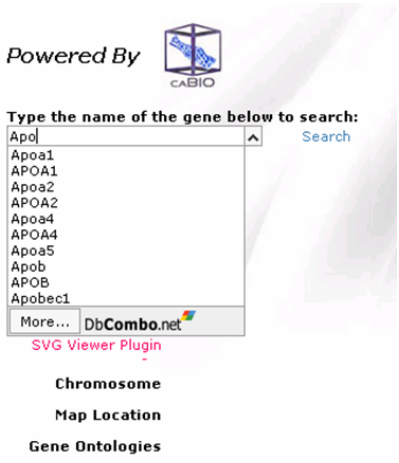



**Figure 7**. Example of AJAX retrieval of genes from the site.

to scroll through the list. We have implemented the AJAX controls for both the strains, genes and markers. This becomes important as we build search facilities for microarray data.

**Added Bioinformatics subsection (data/object models, software distribution, Webservice WSDL file).** During this last cycle, we have had a number of people contacting us regarding the informatics infrastructure. We decided to create a bioinformatics subsection of the web portal to provide the community with the details of the site, the database schema and the object model. We also wanted to create a way for us to provide the software we developed for the consortium that to the public. The bioinformatics subsection provides the public with access to an explanation of the details of the data and object schemas. All software is provided under an Open Software License v1.1 and allows all the source code and executables available to the public for any software we create (e.g. caBIONet). In addition, we also provide an explanation of the web services available, a sample application using them and the WSDL files that describe the services available from the portal.

**Added Annual Reports and SC Presentations subsection.** As part of the open access policy of the AMDCC, we have added sections to the web portal to make the presentations and annual reports generated by the consortium available to the general public. The presentations are from the semi-annual steering committee meetings while the Annual Reports are produced every March and presented to the EAC during the Spring Steering Committee meeting. Because of file size considerations, each PowerPoint presentation and Annual Report is converted to an Adobe PDF document and uploaded to the server.

**Improved Member Page flexibility for users.** As the AMDCC CBU, we are constantly trying to increase the flexibility and user experience on the web portal. Members of the AMDCC are presented with a synopsis of their data, laboratory information and manuscript information upon logging into the system. This information is presented in the context of a number of modules along the sides of the member home page. The position of these modules was originally hard coded, but the membership wanted a little more flexibility. We created a java based drag and drop system for the modules so the members can re-organize these modules. The positions of the modules are saved in a cookie on the user's computer so the next time they visit the page the look and feel will be maintained. Figure 8 is a screen capture of the drag operation.


**Figure 8** Example module drag-and-drop operation to customize the locations of the various modules in the Member Home Page.

**Completed FAQ framework and training modules for the public/membership.** One of the goals for this year was to improve the training and frequently asked questions aspect of the portal. Previously, we had these sections for the membership, but did not have a general FAQ or training videos for the public. During this last year, we completed a number of training videos and created an XML based FAQ system. The portal allows the public to scroll through the list of FAQs and administrators can add/edit new FAQs directly on the portal. Figure 9 is a screen shot of the FAQ page we have created for the portal.

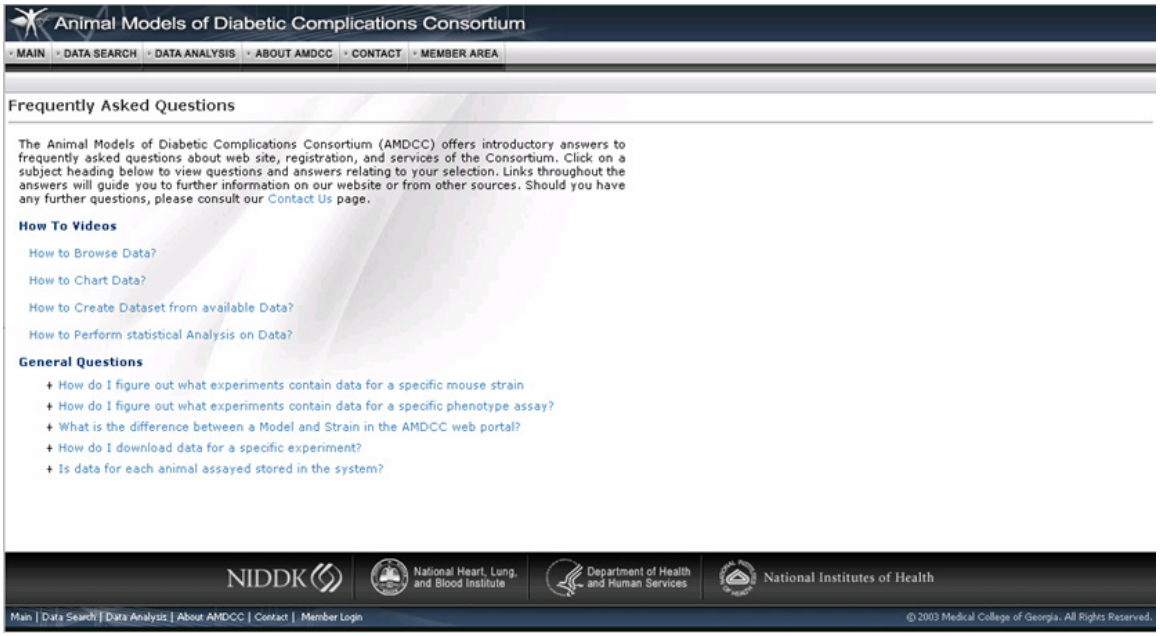



**Figure 9.** Screen shot of the AMDCC FAQ page. This page contains both training videos as well as the general FAQ section.

**Microarray Visualization and Analysis.** We recently completed the Analysis Engine Framework (see above) and are now working on the user interfaces to search, retrieve and analyze microarray data that has been uploaded to the system. The search facilities take advantage of AJAX to dynamically fill dropdown and tree based menus based on the selection of other menus. This prevents excessive posts to the server to fill the dropdowns. Instead, the client browser contacts the server asynchronously and requests the information based on the client controls being selected. For example, if you choose a specific array design, the search facility will automatically list all arrays in the system that have the same design. Figure 10 is a screen shot of the current state of the Microarray search page. Please note this is the preliminary page and displays data in our development servers. In addition, we will provide intelligent filtering criteria. For example, production of volcano plots to visualize cut offs for intensity and p-values to be used in further analyses.

To speed up the creation of visualization pages, we are using the built in COM libraries from S-PLUS to connect to the statistics server, perform the analysis and create the visual java control to display on the page. This is accomplished in our framework by providing a wrapper analysis class that performs all the connections from the compute nodes.

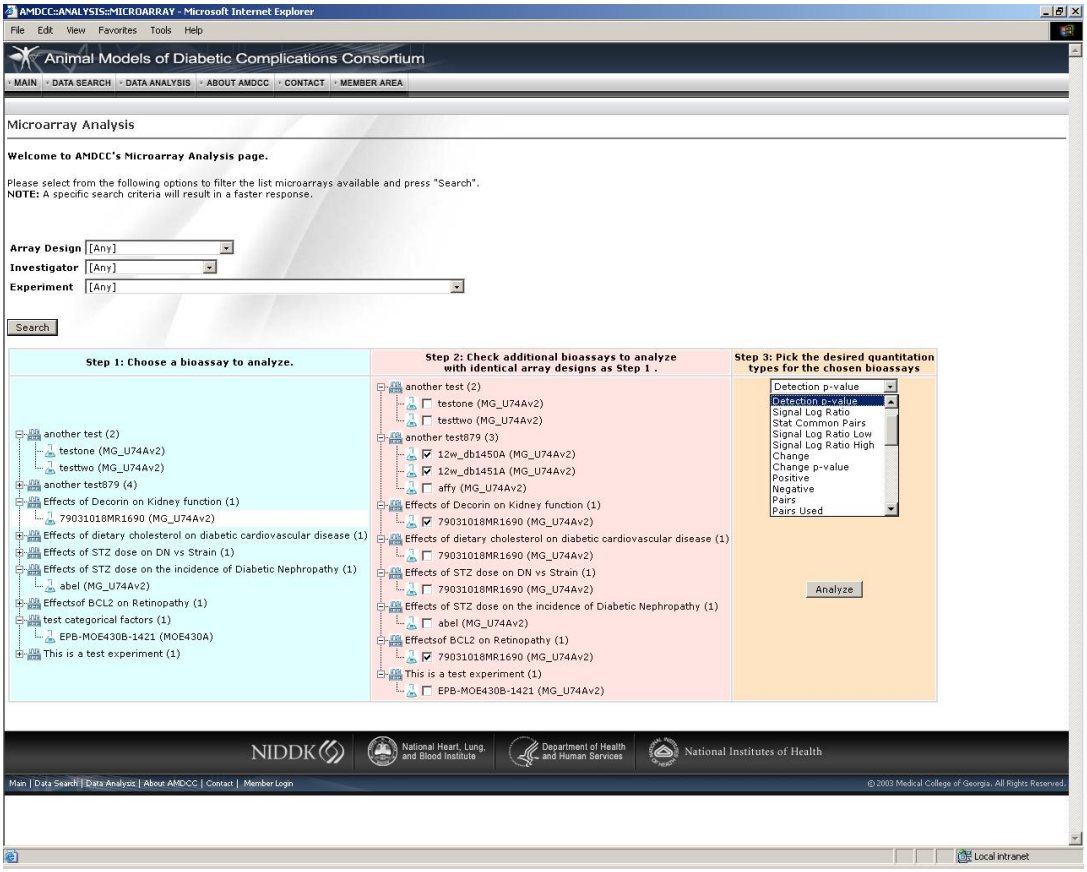



**Figure 10.** Screen shot of example Microarray Search page used prior to analysis. Arrays are listed by experiments and subsequent calls to the server are done via AJAX. Please note this is running on our development servers and is presenting data from the development database.

## 2. Address previous EAC comments:

- *Existing strain kidney weights should be uploaded to the Mouse Phenome database. Are there other existing datasets that could be uploaded?***.**

  We will contact the Mouse Phenome Database (MPD) administrators to see if that is feasible. It may be that the data will have will need to be uploaded under a specific experimental protocol. The investigator that measured the kidney weights for the MPD used a specific experimental design. Our data was collected under different experimental designs. For example, the kidneys were weighed after a number of weeks post STZ induction of diabetes. We should be able to add the data, but it may take some time to organize with the MPD.

- *The website should establish minimum strain/phenotypic data for all mice uploaded to the website. You are strongly encouraged to make sure that these criteria match well with existing databases (e.g. the Mouse Phenome Database, the Complex Trait Consortium, etc.)*

We would like to discuss with the EAC the details of what they mean by minimum strain/phenotypic data for all mice. Does this mean, for example, that all experiments should include a blood glucose measurement for each of the mouse data points? If so, does this mean that if a time course experiment is performed, each time point/animal combination a blood glucose measurement would be enforced? These kinds of policies would have to be debated and voted by the Steering Committee. Once a policy is established, we can certainly enforce it during data upload.

.