ACCESS AND DISPLAY of LoanSTAR DATA VIA THE INTERNET

LoanSTAR Deliverable Report

Final Report

James Sweeney, Jr.
Dan Lockhart
Jeff S. Haberl, Ph.D., P.E.

December 2001

PREFACE

This report is one of a series of reports that documents the development of the LoanSTAR and Technical Assistance deliverables. The developments are broken down into two divisions, Task C and Task D. The following two tables itemize the deliverables for each Task. The information included in this report represents the LoanSTAR and Technical Assistance deliverables Task C - Numbers One and Four.

Task C Deliverables	Description
1.Access and Display LoanSTAR Data via the Internet for Interested Agencies	Energy Systems Lab's web-based interface to the LoanSTAR Energy Databases, referred to as the Energy and Environmental Data System (EEDS).
2.Report Print/ Viewing Options (i.e., PDF format)	Effort to move reporting to Adobe Inc.'s Portable Document Format (PDF).
3.Online Feedback Ability (automated email to ESL)	Online feedback form built into the ESL's EEDS.
4.Internet Data Logging and Display a.Develop Internet-Based Real-time Display Software b.Buy/Develop and Test a Date Logger with Integrated TCP/IP Connectivity c.Buy/ Develop and Test Software to Poll a TCP/IP Logger	Research and development of internet based power measurement and energy data logging techniques.

Task D Deliverables	Description
1.Automated Email Polled Data Problem Alerts. (ESL internal)	The automatic notification via email of the sites that failed polling for a given polling job.
2.Automate IPNs Review Process to an 'Exception Only' Basis (ESL internal)	Streamlining the IPN review process by improving problem tracking and reporting.
3.Convert IPNs plots to PDF format (ESL internal)	Effort to move IPN reporting to Adobe Inc.'s Portable Document Format (PDF).
4.Automated the Data Analysis/Quality Verification Process to 'Exception Only' (ESL internal)	A series of checks that monitor raw files loaded into the database for low level data quality analysis.
5.Move IPNs, MECRs, AECRs to an Internet Base	Effort to combine reporting in Adobe Inc.'s Portable Document Format with a secure Web-based interface.

EXECUTIVE SUMMARY

The information included in this report represents the LoanSTAR and Technical Assistance deliverables Task C - Numbers One and Four, "Access and Display LoanSTAR Data via the Internet for Interested Agencies" and "Internet Data Logging and Display". The material included in this report represent the work of several individuals at Texas A&M University's Energy Systems Lab, namely Mark Simoneau, James Sweeney and Ben Sommers, as well as review and comments by the LoanSTAR Principle Investigators.

ACKNOWLEDGEMENTS

This project would not have been possible without the support that was provided by the following persons and/or the agencies or companies for which they work: Ms. Theresa Sifuentes and Mr. Dub Taylor at the Texas State Energy Conservation Office.

TABLE OF CONTENTS

1.0 INTRODUCTION

The Energy and Environmental Data System (EEDS) is a web application that facilitates web-based viewing of the Energy Systems Lab's extensive building energy databases. EEDS is the initial framework for a growing set of internet energy analysis tools.

2.0 ENERGY AND ENVIRONMENTAL DATA SYSTEM (LoanSTAR Online)

The LoanSTAR Online portion of EEDS is a system comprised of two primary components, the Data Tools component and the User Account component. The Data Tools component provides the user a collection of energy data visualization functions. The User Account component allows the user to tailor data display to his/her personal preference. The following diagram portrays the EEDS structure, components and functions.

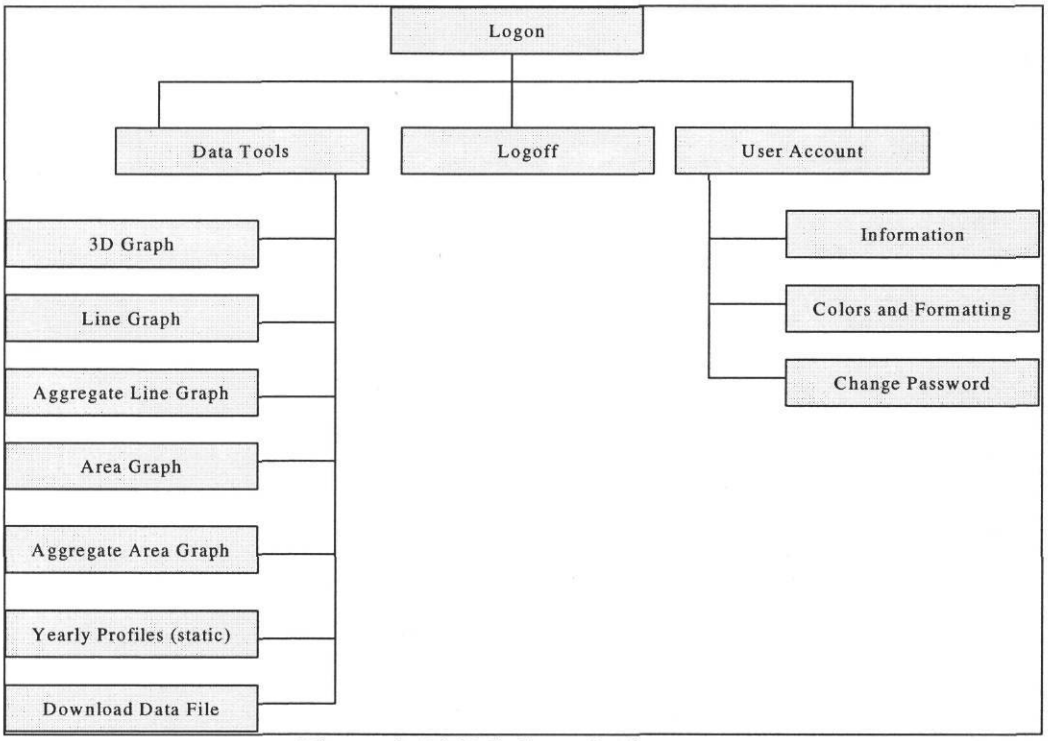

Figure 1. EEDS Generic Structure.

Figure 1 displays the generic structure or "site map" of the EEDS web application. This report discusses the overall use of the EEDS. Some specific tools are discussed in more detail.

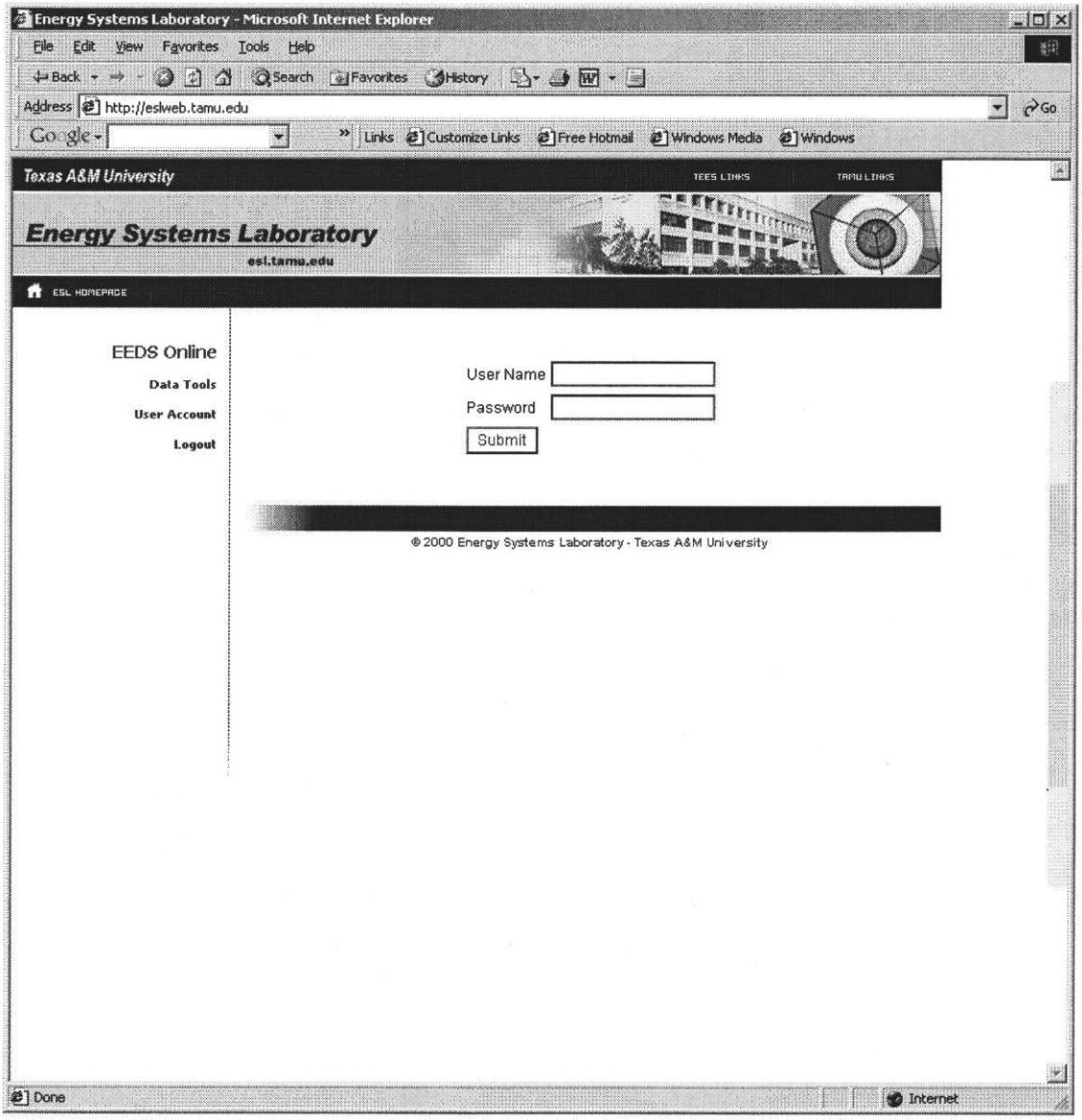

Figure 2. Opening screen for LoanSTAR Online (logon page).

LoanSTAR Online (http://eslweb.tamu.edu) opens up with a logon page (see Figure 2). Here the user enters his/her user name and password for authentication and proceeds to the "logged on" page (Figure 3).

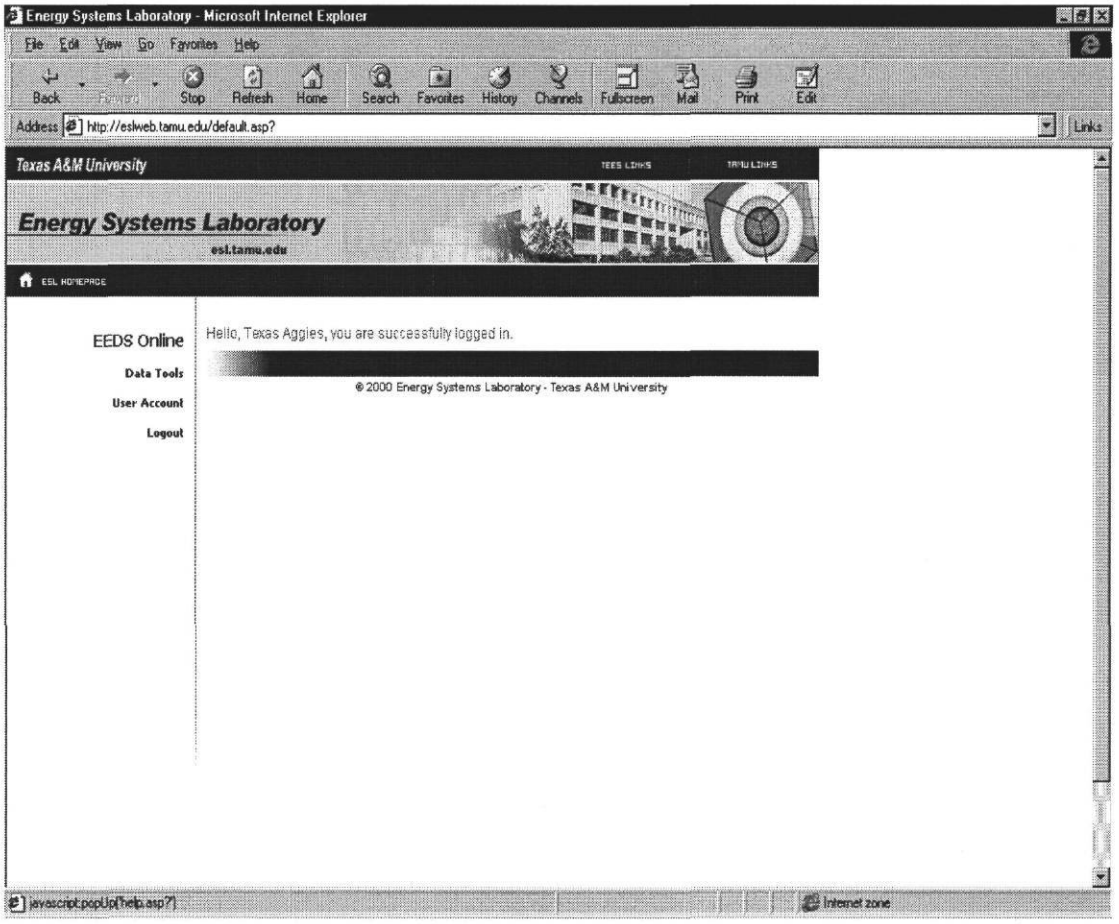

Figure 3. User Logged on Screen.

Figure 3 displays the user logged on to LoanSTAR Online. The panel on the left facilitates access to Data Tools, User Account and user Logout. Clicking on "Data Tools" takes the user to the EEDS tools that can be used to access energy data.

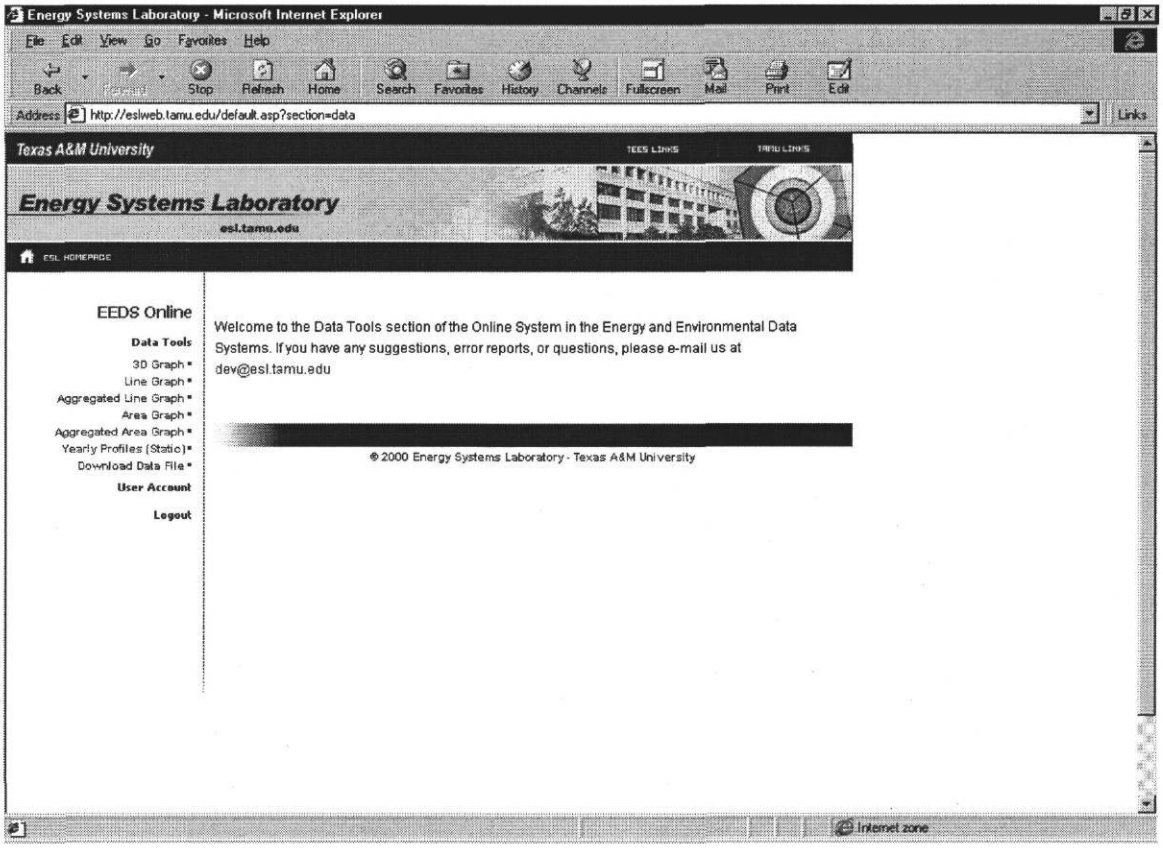

Figure 4. Data Tools Interface.

Figure 4 is a view of the Data Tools section and options to visualize energy data. To view the energy data the user may pick on a variety of tools. For electric consumption and peak demand the user may click on the Line Graph. A listing of the channels available for graphing is retrieved and displayed (Figure 5).

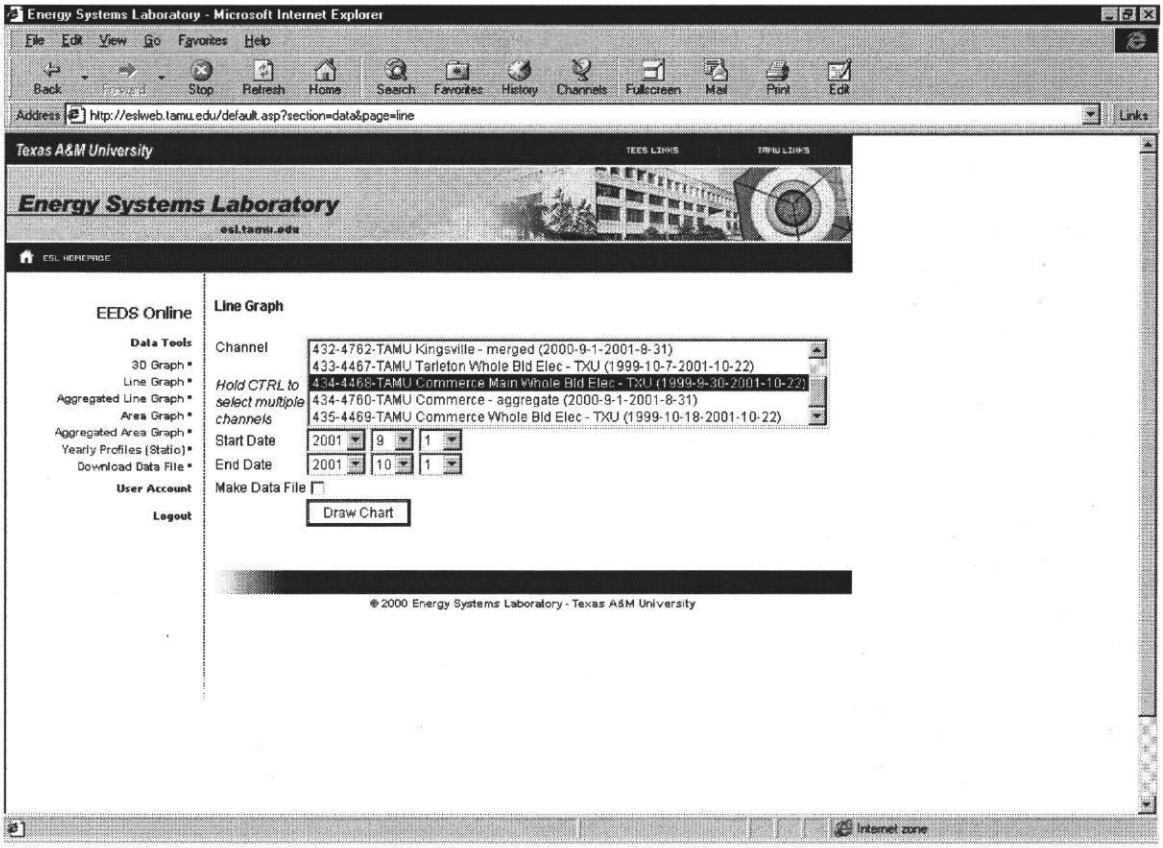

Figure 5. Line Graph Tool Interface.

Figure 5 displays the selection of energy channels to plot. The user clicks on the channel he/she wishes to view. Multiple channels can be viewed by holding down the control key and clicking on one or more channels. Start and end date and times may be changed if the default values are not appropriate. The graph can then be viewed by clicking on the Draw Chart button. A raw data file can also be generated at this point by clicking the "Make Data File" option box.

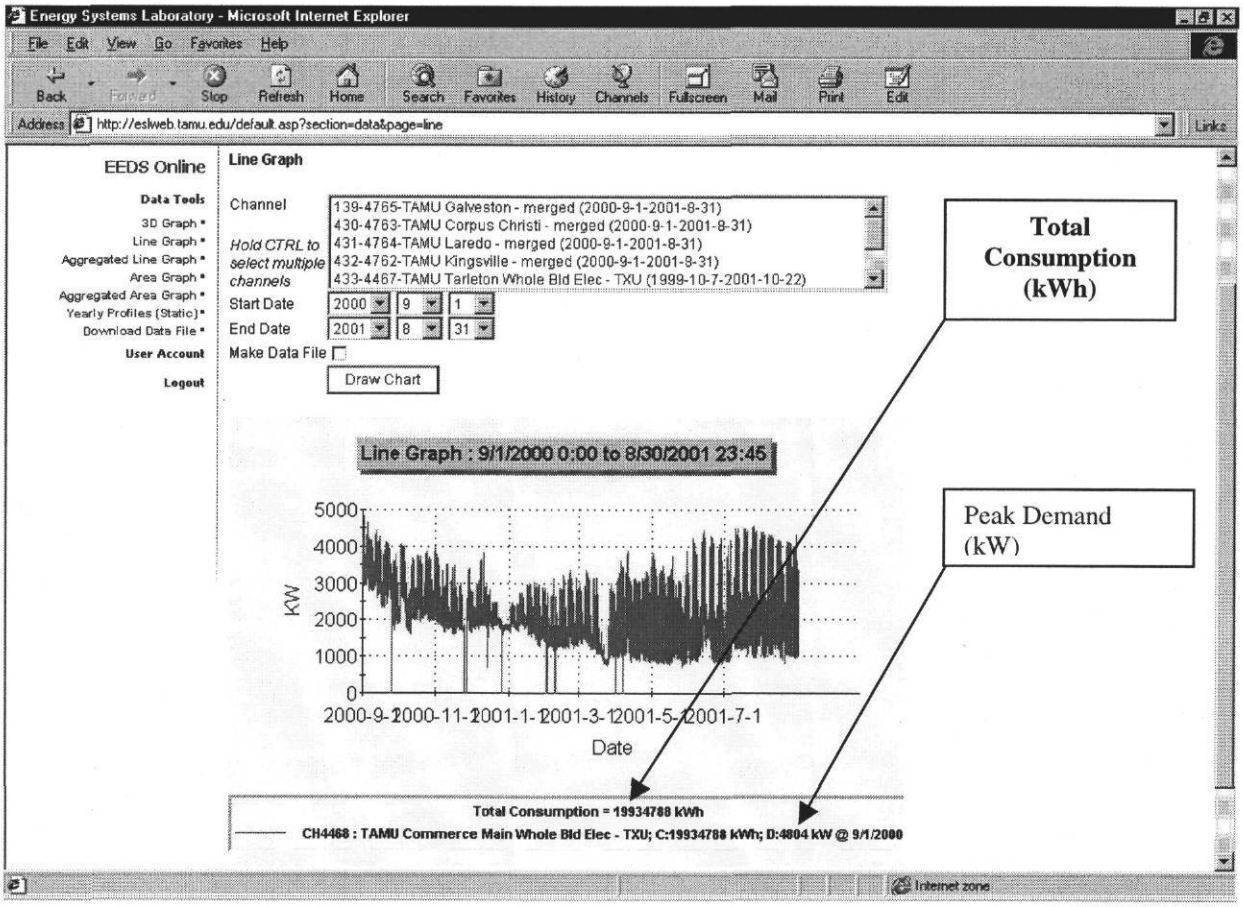

Figure 6. Line Graph plot of Site 434 (TAMU Commerce), Demand vs. Time for the time period September 1, 2000 through August 31, 2001.

In Figure 6 the Whole Building Electricity consumption (kWh) for TAMU Commerce is plotted for one year. The total consumption (kWh) and peak demand (kW) are listed at the bottom of the graph. If the user wishes to aggregate multiple channels, he/she may use the Aggregated Line Graph tool.

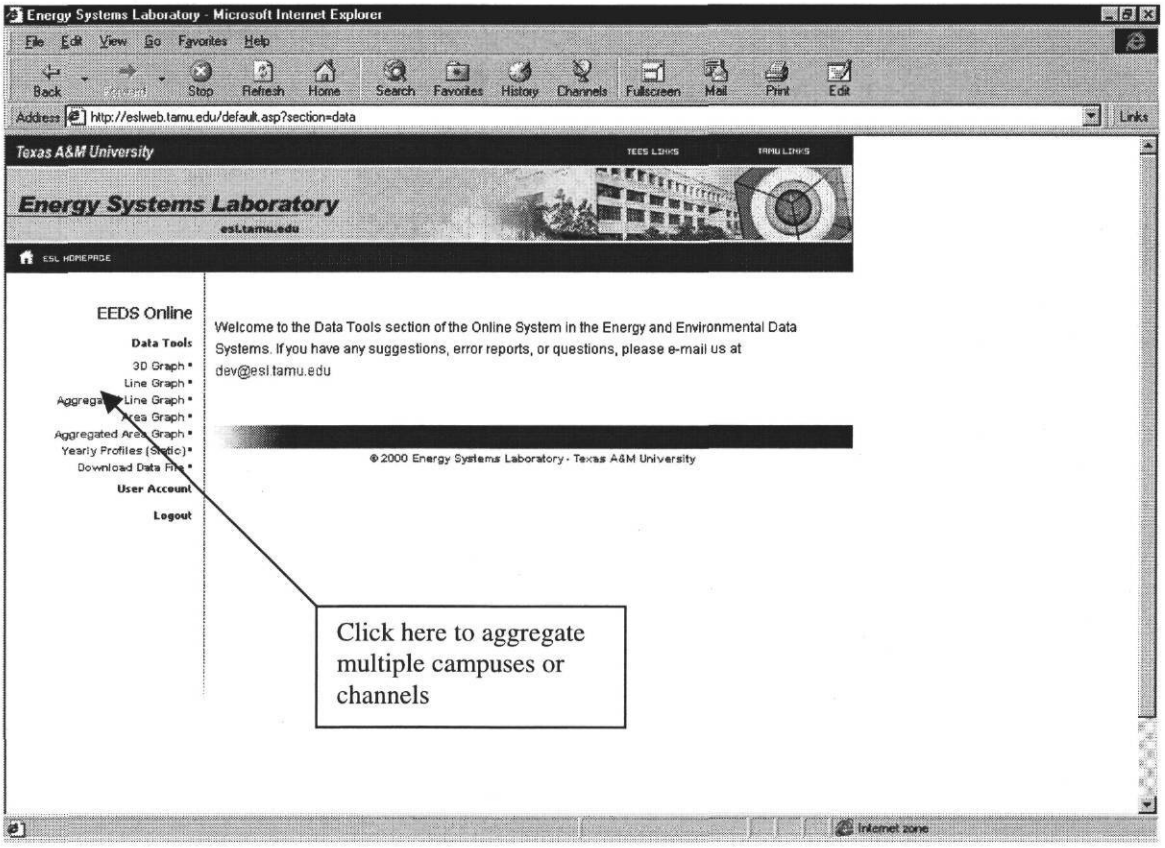

Figure 7. Position of Aggregate Line Graph on the Data Tools Panel.

Figure 7 displays the location of the Aggregate line Graph. It is in the left panel under Data Tools.

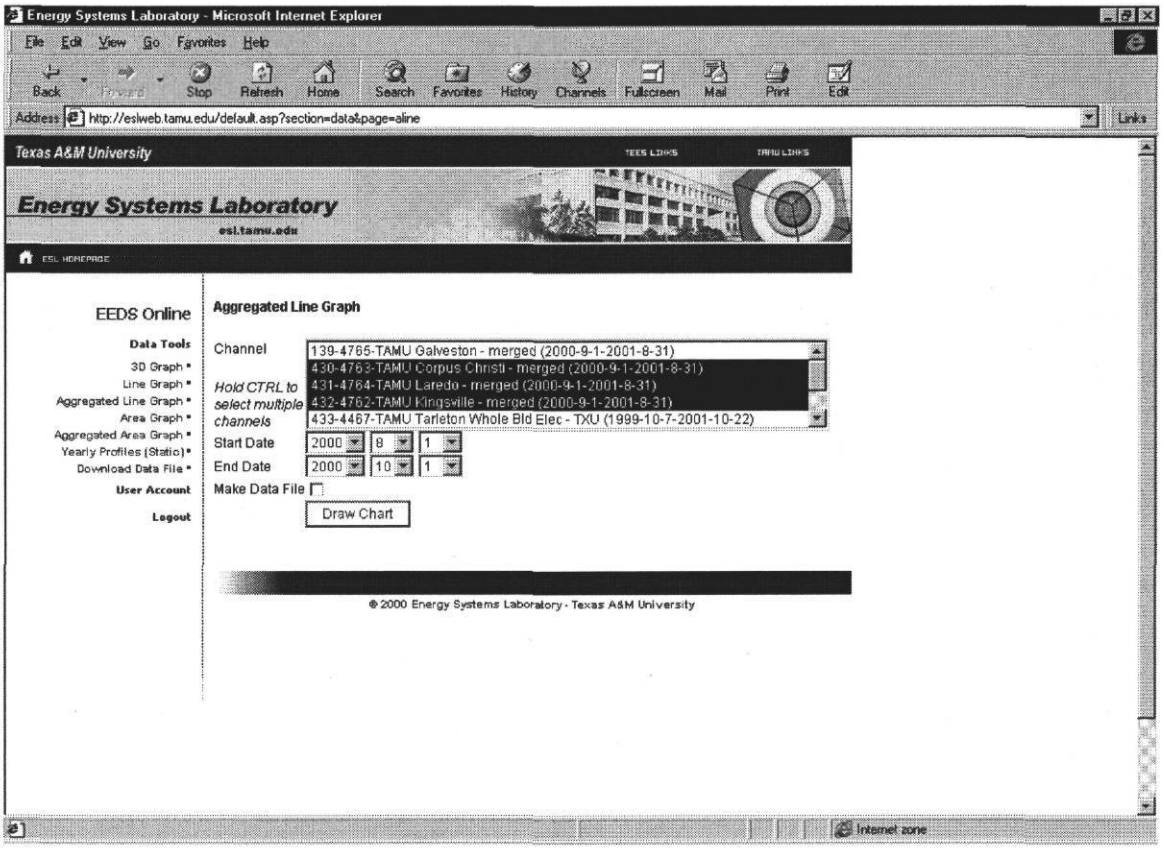

Figure 8. The Aggregate Line Graph Tool Interface.

Figure 8 displays the Aggregate Line Graph Tool. This tool gives the user the ability to stack individual line graphs on top of each other, and yields and aggregated total for all the channel selected. The user selects multiple channels and then draws the chart.

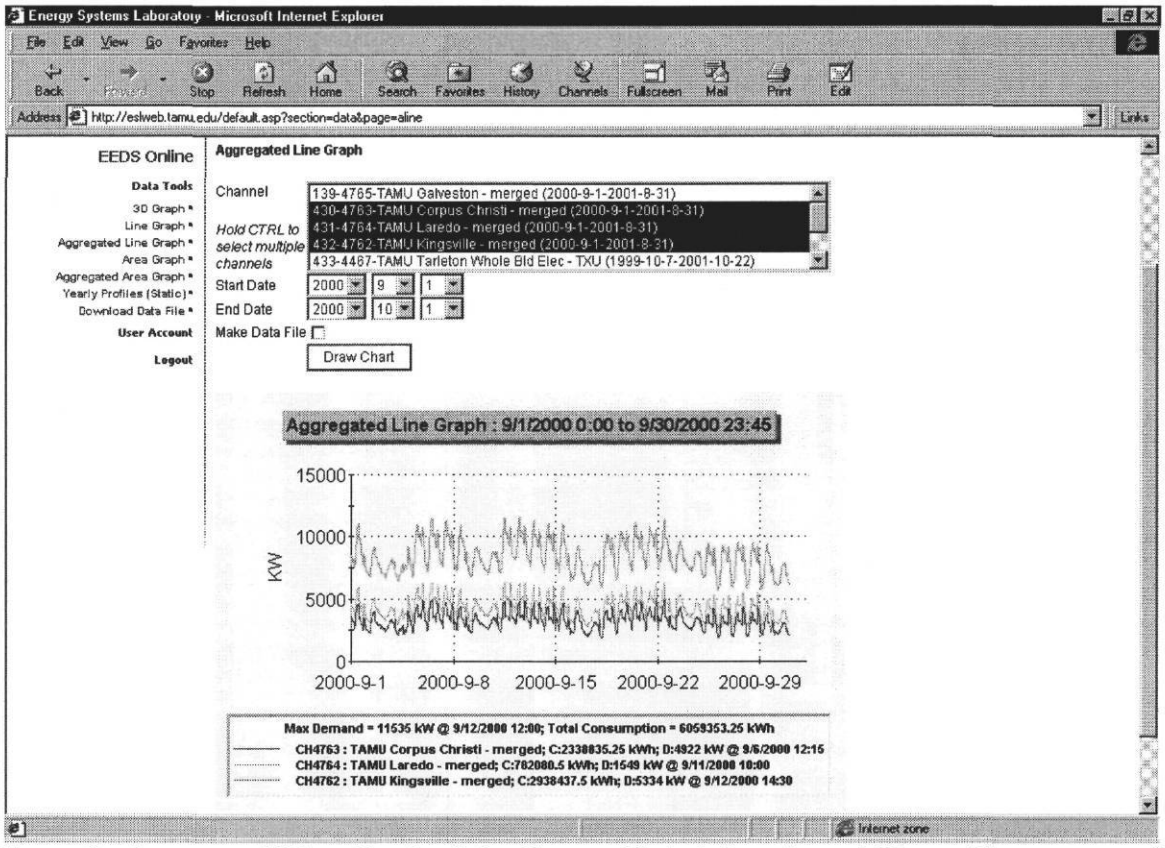

Figure 9. Aggregated Line Graph of the Corpus Christi, Laredo and Kingsville campuses for the time period September 1, 2000 through October 1, 2000.

Figure 9 displays an Aggregated Line Graph for TAMU Corpus, Kingsville and Laredo campuses. The individual channels have been aggregated from bottom to top, yielding an aggregated total of the three channels.

3.0 SYSTEM DOCUMENTATION

3.1 Introduction

This section provides a system level overview of the EEDS application framework, including a functional overview and interface and method detail.

3.2 Functional Overview

3.2.1 3D Chart

Input variables:

- Channel to plot
- Start and End Date
- X, Y, and Z- Rotation
- Legend Toggle

Output

- JPEG image of chart

3D Chart is relatively simple. It takes the input data (retrieved through calls to various Utility functions and other inputs on a form) and passes the StartDate, EndDate and Channel info to the GetData Class(2.1). Once the data is retrieved from the database and placed in Tab-delimited strings, it is passed to the Chart3D Class(2.2) and the Draw method is called. The filename of the generated JPEG is stored in a temporary variable and the popup image is generated by setting the new size of the chart, and calling the Draw() method again. Then both of the filenames are placed into a string that displays the smaller image on the screen and allows a popup window (containing a larger image) to come up when you click on the smaller image.

3.2.2 Line/Area

Input Variables:

- Channel(s) to Plot
- Start and End Date
- Make Data File Toggle

Output

- JPEG image of chart

All of the Line/Area functions are fundamentally the same (the only difference is a chart type and some labels).

After the input is retrieved through the form, GetData is called several times (depending on how many channels are selected). ChanInfoEx is called for each channel to get the vital information regarding sampling intervals, units, server, etc. Then SetConnectString

is called based on the server, SetChannel is called to set the proper table to retrieve data from, and SetInterval is called to set the proper interval for sampling. SetScale is called for conversion purposes. It uses GetRatio from the Utils(2.4) file. After all of that is set, GetData() is called and the resulting data values are stored in an array element for passing to the drawing function.

The OChart2D Class properties are set via Init and SetMultipleCaptions and the Draw method is called. The filename of the generated JPEG is stored in a temporary variable and the popup image is generated by setting the new size of the chart, and calling the Draw() method again. Then both of the filenames are placed into a string that displays the smaller image on the screen and allows a popup window (containing a larger image) to come up when you click on the smaller image.

3.2.3 Yearly Profiles

Input Variables:

- Channel to Retrieve [Limited to Profile Channels]

Output

- JPEG image of chart

The yearly profiles are very simple. The database stores corresponding channel numbers and Yearly Proflie JPEG images. The channels are retrieved by checking to see which channels have corresponding JPEGs and which, of those, the user has access to. After getting the input, the page simply takes the filename in the database (the actual image will eventually be stored here) and sets a link to DBImage.asp where the image is retrieved (either through the file system or the database) and displayed.

3.2.4 Download

Input Variables:

- Channel to Download
- Start and End Date

Output

- Text document containing the categories and corresponding values

Once the input is retrieved, a GetData instance is created and initialized. SetInterval and SetScale are called to properly convert the data. A OChart2D Class is then created and SetData1D is called. After MakeDataFile is called, a link to the file is created.

3.3 Interface details

3.3.1 GetData

Getdata is the class that does all of the interaction with the database when getting data from channels (i.e. chXXXX tables). It properly scales the information, ensures that there is information or NULL values from the start to the end dates given, and promptly return guaranteed valid data. It can convert from any interval to another interval and can have averages or totals if the second interval is larger.

Methods:

`Sub Init (cs, st, et, dfm, cc, vc, dtr)`
This subroutine takes as arguments the Connection String, the Start Time, End Time, Date Format, Category Column, Value Column, and Data Channel. This is simply an easy way to initialize all of the elements of GetData quickly and easily.

`Sub SetConnect (cs)`
This initializes the Connection string seperately

`Sub SetScale (vs)`
This sets the scale to multiply all the data by when converting from one data type to another (See GetRatio() in the Utilities clause)

`Sub SetDates (st, et)`
This sets the start and end dates (rarely used)

`Sub SetChannel (dtr)`
This changes the Data Retrieval channel (used in multiple line graphs)

`Sub SetInterval (i, a)`
This sets the Interval sampled (i.e., 15 minute, 60 minute). If this is equal to the actual interval then there is no change. If greater, then you can specify if you wish to "average" or "total" the intermediate values (this is the "a" argument). If it is less then NULL values are placed in between real values.

`Sub SetNullValue (n)`
This sets the value that will be used if there is no entry in the database or a NULL in the Value Column.

`Sub GetData ()`
This is the actual retrieval function. Calling it invokes calls to the database and places the resulting data in two strings that are accessible through the "Values" and "Categories" properties (described below).

Properties:

`Values`
Returns the string of values (Tab-delimited).

`Categories`
Returns the string of categories (timestamps, typically) (Tab-delimited).

3.3.2 Chart3D

Chart 3D is a class that takes the data from GetData and plots it in a 3D graph with days on the X axes, hours on the Y axes and the energy values on the Z. Different methods allow you to modify the JPEG size, rotation, or any other attributes without having to change the data.

Methods:

`Sub Class_Initialize()`
This subroutine simply initializes the class objects like the labels and captions when the object is created.

`Sub Init(dv, xs, ys, xd, yd, zd, pr, pd)`
This is explicitly called and initializes the data values (dv), the x ans y size (xs, ys), the rotation on all axes (xd, yd, zd), the perspective (pr), and the period or interval that it's being sampled at (pd).

`Sub SetData(dv)`
This simply sets the data string.

`Sub SetSize(xs, ys)`
Sets the x and y size.

`Sub SetOrientation(xd, yd, zd, pd)`
Sets the X, Y, and Z degrees as well as the perspective.

`Sub SetCaptions(tc, bc, lc)`
Sets the Top, Bottom, and Left Captions.

`Sub SetLabels (xl, yl, zl)`
Sets the X, Y, and Z labels.

`Sub SetLegendOn()`
Turns the Legend on or off (if on, generates other information in the Draw() routine).

`Sub SetPeriod (p)`
Sets the interval at which sampling occurred.

`Sub Draw()`
Uses Olectra Chart 3D © to generate the JPEG image of the data passed in using the specified parameters. The JPEG can be accessed through the Filename property.

Properties:

`Filename`

The filename of the generated JPEG.

3.3.3 Chart2D

Chart 2D is a class that takes the data from GetData and plots it in a 2D graph in a given format (Stacked Area, Stacked Line, Plain Area, Plain Line). Stacked charts serve as a summation of the series on the chart, whereas plain charts plot multiple channels for direct comparison.

Methods:

`Sub Class_Initialize()`
This subroutine initializes the class when the object is created (currently, it only turns the legend off.

`Sub Init(dc, dv, tc, lc, bc, xs, ys, ct, pd)`
Explicitly called initialization. Data Categories (timestamps), Data Values (array of strings of corresponding values), Top, Left, and Bottom captions, X and Y sizes, Chart Type (i.e. Aggregated or Not, Line or Area), and Period (Legacy, not used).

`Sub SetData(dc, dv)`
Accepts a string of "categories" and an array of strings of "values" to be plotted.

`Sub SetData1D (dc, dv)`
Accepts a string of "categories" and a string of "values" – used when only one series is needed.

`Sub SetSize(xs, ys)`
Sets the size of the JPEG to be generated.

`Sub SetCaptions(tc, lc, bc)`
Sets the top, left, and bottom captions.

`Sub SetMultipleCaptions(rc)`
Turns the Legend on, and splits "rc" into an array of labels (expected to be a comma delmited list).

`Sub SetType (ct)`
Sets the chart type.

`Sub SetPeriod (p)`
Legacy, not used.

`Sub Draw()`
Takes the data values and all the parameters given and plots them on a graph

using Olectra Chart 2D © and generates a JPEG image. This can be accessed through the Filename property.

Properties:

`Filename`
The filename of the generated JPEG.

3.3.4 Server Side Methods (VBScript)

`sub CleanObject(ByRef objObject)`
Error proof cleanup of an object (closes the object and sets it to nothing).

`sub ErrorHandler (byref objPreferences)`
Catches errors and logs it in database.

`function AllocateTemporaryFile()`
Allocates a temporary file (generating JPEG's, TXT files, etc).

`function FormatDigit(intNumber, intTotalDigits)`
Returns a number as a string of length 'intTotalDigits' by adding zero's to the beginning (good for channel formatting and dates).

`function FormatDate(ByVal dtmDate, ByVal strFormat)`
Makes the date 'dtmDate' into a string in format 'strFormat' (which can contain "YYYY", "MM", and "DD".

`function GetRatio(strFromUnits, intFromInterval, strToUnits, intToInterval)`
Returns a scale for converting the FromUnits to the ToUnits.

`function GetElement(ByVal strName, ByVal intType, ByVal varDefault)`
Returns an element passed to the page. If the element does not exist or is not of the specified type, it returns 'varDefault'.

`sub FormDate (ByVal strControlName, ByVal dtmInitialDate, ByVal strFormat)`
Makes a control in a form that uses Javascript to return a date in one hidden form value.

`sub FormChannels (ByVal strControlName, ByVal strInitialValue, ByVal blnIsMultiple, ByVal strFormat, ByVal strDateFormat)`
Makes a control in a form to allow the user to select one or more channels that they have access to.

```
sub ChannelInformation(intChannel, intInterval, strUnits,
    strServer, strDescription)
```
Returns information for a specific channel from the database.

Client Side Method (JavaScript)

```
function MessageBox(loc, xsize, ysize)
```
This has some bugs, currently (it was once 4 different functions), but will eventually.

4. 0 FUTURE DIRECTIONS

Other tools and usability components will be added to the EEDS as funding permits. The following components are considered as high priority items:

4.1 Bar Chart

The Bar Chart will involve a new functionality for OChart2D. It will graph demand and consumption over various intervals and varying time periods.

It will take as input :

Channel(s) to plot
Start and End Date
Time Period to group by : Hourly, Daily, Weekly, Monthly, etc.
Demand or Consumption Toggle

And will return a bar graph that has N groups of bars (where N is the number of time periods within the Start and End date range (ex : 24 for 1/1/01-1/2/01 set on an Hourly Period)) that either have the consumption of that period or the maximum demand for that period.

4.2 Channel "User Sets"

Channel sets are a feature to be added in the future that allows users to have predefined sets of channels to grab with predefined graph types. The basic idea is to have a location on the site where users can select channels, a graph type, and the independent variable to graph from (TIME or a particular channel).

4.3 Inspection Plot Notebook (IPN) Functionality

This would allow a user to put multiple graphs (of different types) on the same graphic to save or print out, similar but more functional that the current IPN. Currently, IPNs are processed in a weekly batch mode and are not very flexible. This tool would allow the Data Inspector to have complete control over his/her Notebook and would be able to load the data on demand.

4.4 **Whole-Building Expressions**

Whole-Building Expressions (WBEs) are currently used in the ESL's Monthly Energy Consumption Report. They allow the user to be more agile when manipulating data. Instead of simply specifying a single channel, a whole-building expression can be an expression that incorporates multiple channels in some type of formula so that the entire building's energy consumption can be accurately depicted. This is particularly useful when looking at aggregate loads, sub-metering, and temperature data. An expression parser and the expansion of GetData method to accept WBEs rather than individual channels.

4.5 **Off-Line Image Generation**

As more people begin to use this system, there will be a large number of people retrieving the same data or consistently requesting predictable data sets. As a result, some services or scripts could be created to grab data and create the images during non-peak hours. This could also be used when retrieving very large amounts of data.

4.6 **Scatter Plots**

The plotting of an energy variable vs. another non-time variable (i.e. outdoor air temperature) is used extensively in the Energy Systems Lab's data quality and analysis. Two-variable "scatter plots" are very useful and a Data Tool that has this functionality is needed.