<u>RUMS</u>
RESOURCE UTILIZATION MONITORING SYSTEM

by

T. Duncan, T. K. Chen, D. Pestal and T. Merport

Memorandum No. UCB/ERL 03/43

November 2003

RUMS
RESOURCE UTILIZATION MONITORING SYSTEM

by

T. Duncan, T. K. Chen, D. Pestal and T. Merport

Memorandum No. UCB/ERL M03/43

November 2003

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

RUMS

Resource Utilization Monitoring System

Microfabrication Laboratory
University of California, Berkeley
July 2003

T. Duncan
T.K. Chen
D. Pestal
T. Merport

ABSTRACT

The Resource Utilization Monitoring System (RUMS) installed in the Microfabrication Laboratory at U.C. Berkeley is comprised of data acquisition hardware, sensors, and a set of software tools. This system is designed to monitor the facility utilities and fabrication equipment. RUMS facilitates environmental compliance, safety, and processing in the Microfabrication laboratory by gathering information from a variety of sensors, applying calibration factors, testing sensor data against limits, notifying personnel for any relevant change in sensor conditions, and archiving the data to a centralized database.

TABLE OF CONTENTS

LIST OF FIGURES

1 *INTRODUCTION*

Facility and resource monitoring in the Microfabrication Laboratory are important to insure its safe and efficient operation. Data is transmitted from a variety of sensors in the Microfabrication Laboratory to a computer based data acquisition system. The data acquisition system uses a 64 channel plug-in board from National Instruments. Data is processed through the principal application software (RUMS Server) developed with the LabVIEW development environment (also from National Instruments). The resulting data are stored in an SQL based relational database and available for viewing, analysis, or configuration through the Web or a specialized LabVIEW application that can be run from Windows 2000/XP workstations (Figure 1.) Database administration programs have been developed to permit database monitoring and data archiving. Additionally, the RUMS Server provides remote access through standard network sockets (through the embedded resource server application). This access allows clients on other platforms to query gas usage or tank levels. Remote access to the RUMS server also permits system monitoring from a remote host.

RUMS replaced an older set of tools for facility monitoring: the Berkeley Laboratory Information Monitoring Program (BLIMP). BLIMP utilized a twenty year old data acquisition platform and an archaic flat file data storage system (which provided an impetus for the development of RUMS). RUMS kept many of the features designed for the BLIMP system that proved useful over many years of service. RUMS also added many features afforded by a modern development environment.

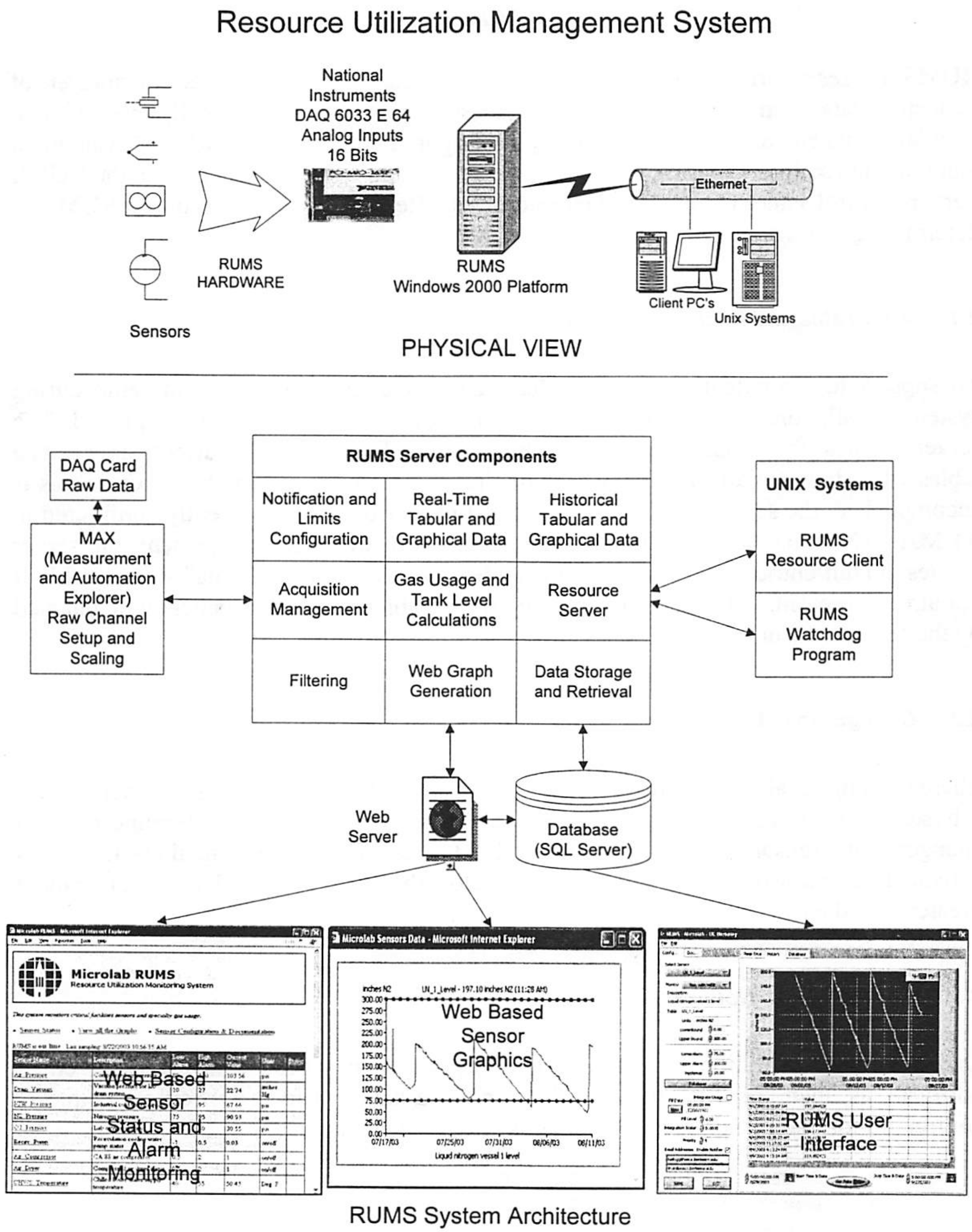


Figure 1 Physical View and Software System Architecture

2 *MAJOR FEATURES*

RUMS utilizes a variety of technologies to simplify configuration, storage and analysis of collected data, and remote access. Database tables are automatically created and populated when a sensor is initially configured using Labview's Measurement Automation Explorer. Alarms, email, and virtual sensors can be configured on the RUMS Server Control Panel or through web interfaces. Remote systems can query RUMS to determine gas usage or tank levels.

2.1 Auto Table and Record Generation

To support to the extent possible the least amount of configuration and setup during system installation, a dynamic method of database table generation is employed. The server creates the Configuration, Documentation, TableNames, CurrentValues state tables when first initialized, if they are not already present in the database. Once this is accomplished, the server checks the current list of Sensor Names presently configured in NI MAX (National Instruments DAQ interface.) For every Sensor present, the server creates default entries as necessary in the tables and also creates a "data" storage table if not already present. All referential integrity of the tables is programmatically maintained by the server system.

2.2 Configurable Alarm Limits

There is an upper alarm and lower alarm limit configurable for each sensor. There is also a hysterisis value associated to each sensor. This value is used to determine relevant changes in the sensor data. If an alarm threshold is reached, optional email notification is activated. The sensor level must then return below the alarm threshold and by an amount greater than the hysterisis value prior to another alarm being activated.

2.3 Optional Email Notifications

Once an alarm condition has been met, if the sensor has been configured for email notifications, then email is automatically generated and sent to the specified recipients configured for that sensor. The following is an example email:

```
9/10/2003           4:34 PM
Sensor              Test_Tank
Description         Test tank
Low Alarm           0.00
Upper Alarm         60.00
Currently           60.24

Just thought you'd like to know,
The Computer Support Team.
```

2.4 Referential Documentation Component

Each sensor has a record in the "Documentation" table which describes the Vendor, Location, Type, Contact, Support email, among other fields. This information is available for quick reference to aid in troubleshooting and maintenance.

2.5 Configurable Database Connectivity Interface

The server initially starts up utilizing an Access database file (*.mdb) to support an integrated database solution. For corporate or enterprise database solutions, one may configure the database connectivity information by editing the rums.udl connectivity information directly with the built-in Microsoft UDL editor dialog. This dialog can be launched directly from the server or client respectively.

2.6 Intelligent Data Storage to Database

During raw sensor data monitoring, the configured hysterisis value is used to determine the noise level of the data. New data records are only stored to the database when the current sensor value has diverged from the previously saved value by the hysterisis amount. For sensors that are integrated to determine total consumption (such as a gas flow rate sensor), new data records are stored every five seconds while the sensor is in the upper alarm state, indicating a flow condition. This allows for an accurate integration process. The integration algorithm is used is the trapezoidal method.

2.7 Virtual Sensors

Along with the standard physically present sensors, a mechanism is employed to implement "virtual" sensors. Virtual sensors are sensors whose data are "calculated" from the data obtained from one or more physical sensors. The current system allows for defining "Tank" virtual sensors and Moving Averaged virtual sensors. Throughout the lab, a number of different tools utilize gases obtained from common cylinders. The gas flowing to the separate equipment is monitored by physical flow rate sensors. In the configuration of these flow rate sensors, one can define whether the sensor is to be "Integrated" and also an "Integration Scalar" can be defined for converting the dimensions and units from flow rates to integrated units of interest. When defining a virtual tank sensor, one can configure one or more of the flow-rate sensors to be used in determining the real-time gas resource usage. This information is used in calculating the remaining gas resource. In this manner, alarm emails can be generated when a certain gas resource is nearing depletion. For the Moving averaged virtual sensor, a "real" sensor can be filtered using a moving average algorithm. The window size of the moving average is configurable for each virtual sensor. This method allows for filtering noisy data that is of no interest, such as the Chilled water pressure which undergoes minor fluctuations that are of no real interest. Other filtered virtual sensors can be implemented readily with the existing infrastructure employed with the Moving Average filter.

2.8 RUMS Resource Server / Client Side

Furnaces in the Microlab utilize germane, disilane, and other critical gases. The RUMS resource server facilitates accounting and processing in the Microlab by calculating gas resource usage or tank levels from flow sensors attached to furnace tubes. Microlab software systems contact the server when a furnace run is completed, query the gas usage, and automatically apply charges to a member's account. Microlab databases store properties that are associated with each furnace tube to determine which sensors to query when the member's run is completed.

2.9 Web Interface

Most day to day interaction with the RUMS system will be through its various web based interfaces. Sensor states can be monitored, graphs viewed, and sensors can be configured through these interfaces.

3 COMPUTING REQUIREMENTS

The system has a number of dependencies that are required to run on one or more computing platforms. There are separate hardware and software requirements, each of which are identified in the following sections.

3.1 Hardware Requirements

The LabVIEW RUMS Server requires an E-Series Data Acquisition (DAQ) Card from National Instruments. Also required is a PC based system that will accommodate the DAQ card. The recommended minimum PC system requirement to run the RUMS Server component with an E-Series DAQ card is a Pentium II Class machine with 300 MHZ Processor, 256 MB Ram, and 20 GB Hard Drive free space for Database (if the database is to be stored locally).

3.2 Software Requirements

To run the RUMS Server software, one must install National Instruments LabVIEW Runtime Engine version 6.1. This software component is installed automatically with the installation program provided with the RUMS Server installation package. The default configuration is to use an access database file for storing data and configuration information, but the RUMS Server can be configured to use any database either on the local machine or across the network.

4 *SYSTEM ARCHITECTURE*

The system architecture starts with facility requirements. Generally sensors are connected to a data acquisition card and configured using National Instrument's configuration tool MAX (Measurement and Automation Explorer). The RUMS server application collects and processes data streamed from MAX and performs further processing. When the processing is completed, data is stored in a relational database. There are several web based tools to allow users to rapidly access data acquired by RUMS and manage the RUMS database (Figure 1). A watchdog program that runs on a UNIX system checks the health of the RUMS system; it will send mail to administrators if a problem is encountered.

4.1 Facility Requirements

The measurable physical phenomena in the lab, states of which are important for day to day operation, are diverse. Just as diverse is the multitude of available collection and transmission methods of these measurements. With RUMS, two types of remote sensor outputs have been selected: 4-20mA current sensors and simple contact closures.

Measurements of continuous values such as pressure, temperatures, levels and flow rates are acquired and transmitted using current-sensors. Current sensors were chosen since voltage sensors would suffer a loss in signal due to the voltage drop caused by the resistance of the long runs of wire necessary for the placement of some sensors.

Besides tracking continuous measurements, it is necessary to monitor the discrete operational status of motors on pumps and compressors, as well as watch the state of switches and relays. Doing this requires no special sensors since it is possible to use the relays, switches, or motor contactors themselves. By looking at the voltage coming from these, you can tell whether the device is open or closed, on or off. The long runs of wire necessary for these contact closures do not affect the readings, since we are not interested in small changes in voltage, only the presence or absence of voltage.

The exceptions to these two types of sensors are the O2_Flow_Rate sensor and the Germane Tank sensor. Due to the close proximity of the oxygen flow sensor to the RUMS computer, a voltage sensor is adequate. There is no physical germane tank sensor; the measurement the "sensor" makes is only a calculation made from the germane flow sensors, which is then subtracted from an upper level that is manually entered into the software, when an empty gas cylinder is replaced. This sensor is considered "virtual."

Figure 2 shows the list of sensors and their configuration applied in the Microlab.

Sensor Name	Description	Sensor Type	RUMS Pair	DAQ Channel	Connection Type
Air _Pressure	Compressed air pressure	current	14	14	single-ended
Drain _Vacuum	Vacuum pressure for lab drain system	current	20	25	single-ended
ICW_Pressure	Industrial cold water supply	current	21	18	single-ended
N2_Pressure	Nitrogen pressure	current	3	1	single-ended
O2_Pressure	Lab oxygen pressure	current	4	9	single-ended
Recirc_Pump	Recirculation cooling water pump status	contact closure	10	12	single-ended
Air_Compressor	CA 88 air compressor	contact closure	12	13	single-ended
Air_Dryer	Compressed air dryer status	contact closure	13	6	single-ended
CHWS_Temperature	Chilled house water supply temperature	current	22	26	single-ended
Cooling_Tower	Cooling tower pump	contact closure	11	5	single-ended
DI_Chem_Tank	DI chemical injection tank low alarm	contact closure	19	17	single-ended
DI_Level	Level of water in DI storage tank	current	6	10	single-ended
DI_Makeup_Flow_Rate	DI water makeup flow rate	current	9	4	single-ended
DI_Resistivity	Resistivity of DI water	current	46	51/59	differential
Disilane_t19_Flow_Rate	Tystar19 disilane flow rate	current	43	48/56	differential
Disilane_t20_Flow_Rate	Tystar20 disilane flow rate	current	45	50/58	differential
Germane_t19_Flow_Rate	Tystar19 germane flow rate	current	42	39/47	differential
Germane_t20_Flow_Rate	Tystar20 germane flow rate	current	44	49/57	differential
Germane Tank	Germane Gas for Tystar Tubes (t19 & t20)	virtual			
HAZMAT_Pull_Alarm	HAZMAT blue hand pull alarms	contact closure	23	19	single-ended
HAZMAT_Toxic_Gas_Alarm	HAZMAT toxic gas detector	contact closure	24	27	single-ended
HF_70	Exhaust hood fan 70	contact closure	15	7	single-ended
HF_83	Exhaust hood fan 83	contact closure	16	15	single-ended
HF_84	Exhaust hood fan 84	contact closure	17	6	single-ended
LN_1_Level	Liquid nitrogen vessel 1 level	current	1	0	single-ended
LN_2_Level	Liquid nitrogen vessel 2 level	current	2	8	single-ended
Makeup_Air_Temperature	Supply air temperature	current	18	24	single-ended
N2_Flow_Rate	Nitrogen flow rate	current	47	52/60	differential
O2_Flow_Rate	Lab oxygen flow rate	voltage	49	54/62	differential

Figure 2 Sensor Input Configuration

4.1.1 Sensor Hardware Interface

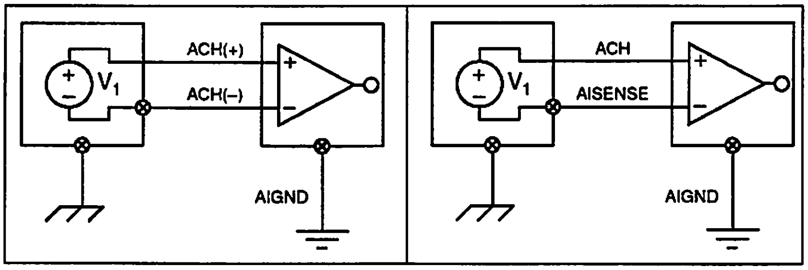


Figure 3 Differential and Single-ended inputs

RUMS employs a National Instruments (N.I.) data acquisition (DAQ) card with 64 analog input channels (NI 6033E). These channels can be implemented in two ways: single-ended, or differential (Figure 3). Sensors that have their own local power supply must be connected differentially; this uses 2 analog input channels (ACH+, ACH-). Differential connections allow the DAQ card to have a reference for the ground of the remote power supply since it is very unlikely that that remote power supply will be equipotential to the ground of other sensors using different power supplies. Sensors that all share a same power supply (or ones that use different power supplies which have the same ground) can be connected to one separate analog channel (ACH), with one ground from the power supply as reference for all the sensors (AISENSE). In this way the DAQ card is able to populate 64 single-ended connections, or 32 differential connections. Fortunately the card has the ability to host both types at the same time considering RUMS utilizes both locally, and remotely power sensors.

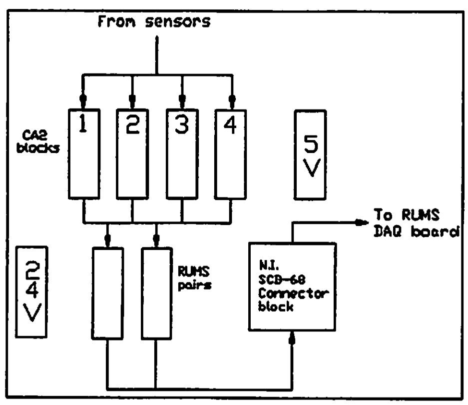


Figure 4 Sensor Wiring Block Diagram

The N.I. DAQ card resides in a PC in CA2 of the Microlab. From this card runs a 100 conductor shielded cable (SH100100) to a N.I. shielded connector box (SCB-100) for low-noise, screw down termination, access to the DAQ channels. From the connector box the channels fan out to two 66M1-50 connecting blocks (RUMS pairs) to supply easy connections for the four 66M1-50 connecting blocks (CA2 blocks) that receive connections from the sensors throughout the lab and building (Figure 4).

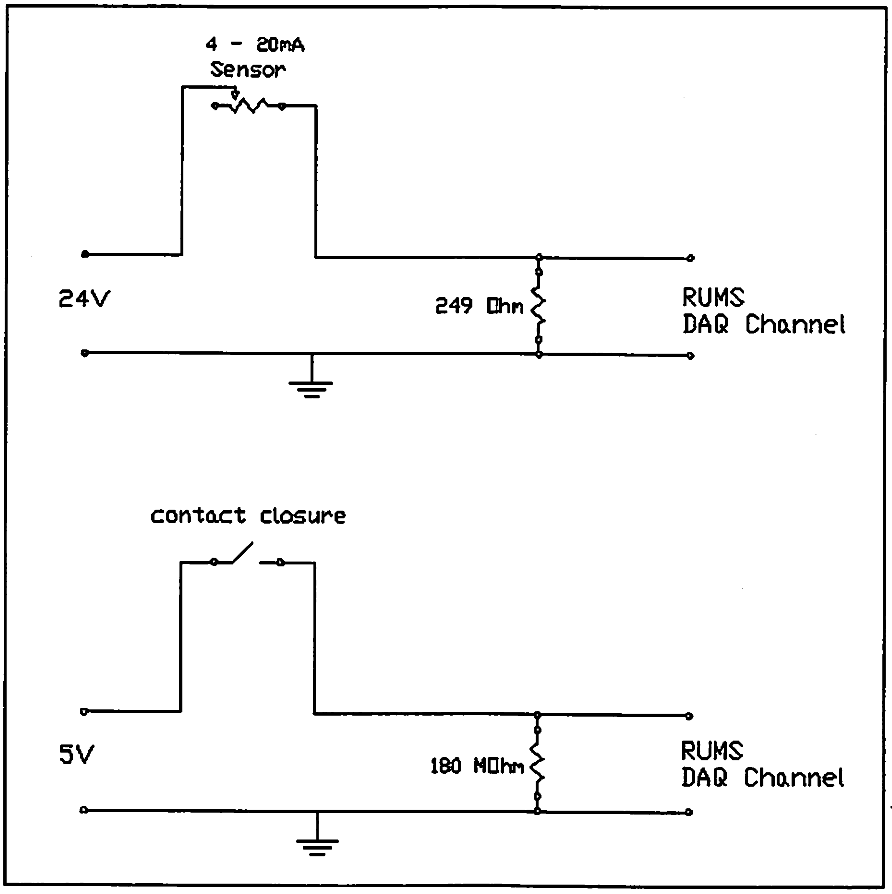


Figure 5 Sensor Connections

24V DC powers all of the 4-20mA sensors. This power is either supplied by a local power supply found near the sensor, or by the 24V power supply located near the RUMS server. All contact closures receive 5V from a power supply also located near the RUMS server. The DAQ card has the ability to take in readings from –10V to 10V DC, which is a problem since the current sensors are working on 24V. To solve this problem the current signal is sent through a 249 ohm, high precision resistor, with the DAQ card reading the voltage across this resistor which is approximately 0V to 5V. Similarly, the contact closure signals are read across a 180M ohm resistor. This functions as a "pull-down resistor" making sure that when the contacts are open, potential left on the floating line is forced down to ground (Figure 5).

4.1.2 Measurement and Automation Explorer (MAX)

Measurement and Automation Explorer (MAX) provides access to the DAQ card on the software side. MAX is used to configure, view, and diagnose the individual DAQ channels. After physically connecting the sensor, the first step in adding a new sensor to RUMS is to configure it into MAX. In MAX, DAQ channels are assigned sensor names, configured to be either current or contact closure (voltage) sensors, and have scales applied to them to correlate the DAQ channel voltage reading to the actual physical measurement of the sensor. After a DAQ channel has been configured, it is then accessible from LabVIEW by the sensor name assigned to that DAQ channel.

4.2 LabVIEW system components

4.2.1 RUMS Server Control Panel

The Front Panel for both the Server and the Client (Figure 6) are identical in both view and user interface functionality. From the front panel one can view or edit configuration and documentation information for each real and virtual sensor. One can view real-time sensor values on a strip-chart graph and in tabular format. One can also retrieve and view historical data in both a graph and tabular format.

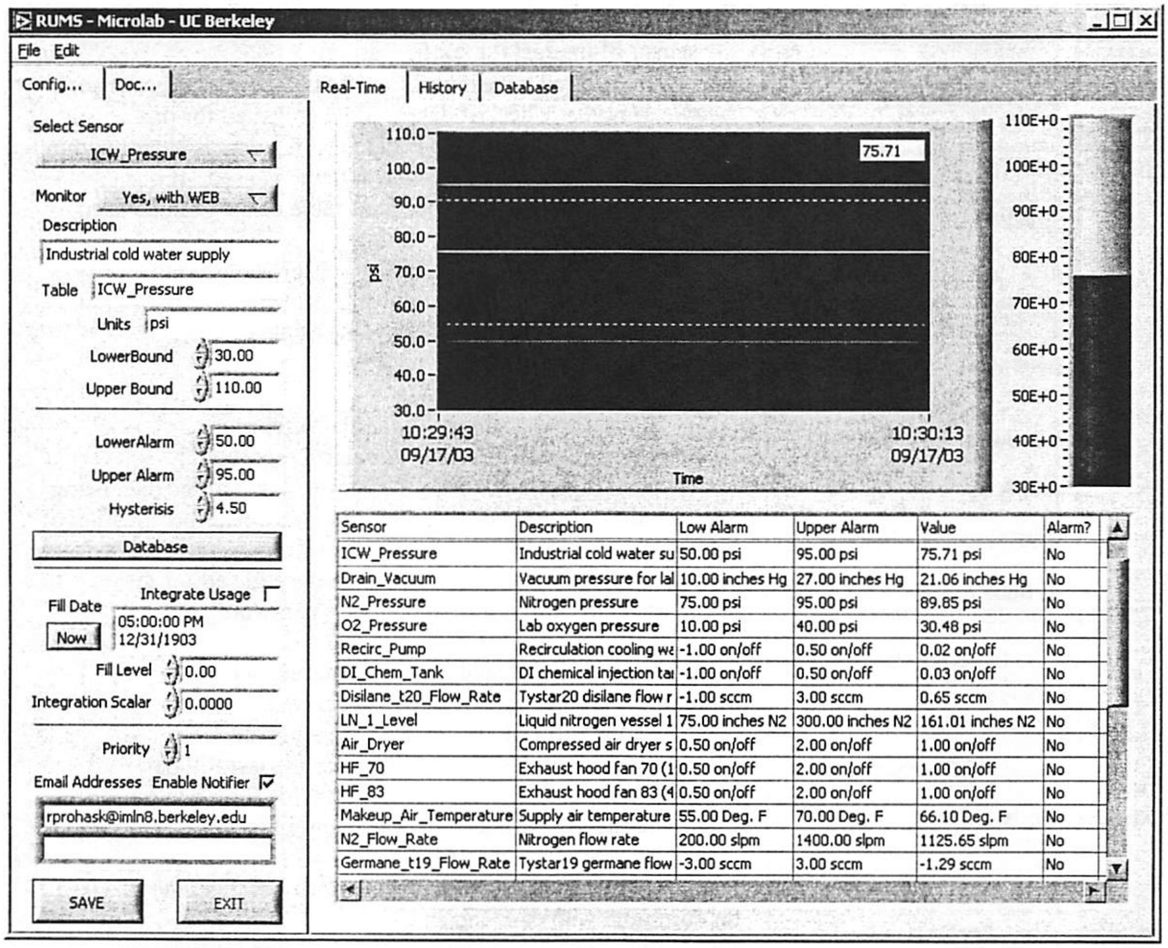


Figure 6 RUMS User Interface

4.2.2 Configuration

Configuration information for each sensor is defined using the configuration sub-panel. The information defines upper and lower alarm limits, upper and lower expected value bounds, whether or not to monitor the data and to display the information on a web graph, a description, and other pertinent information (Figure 7).

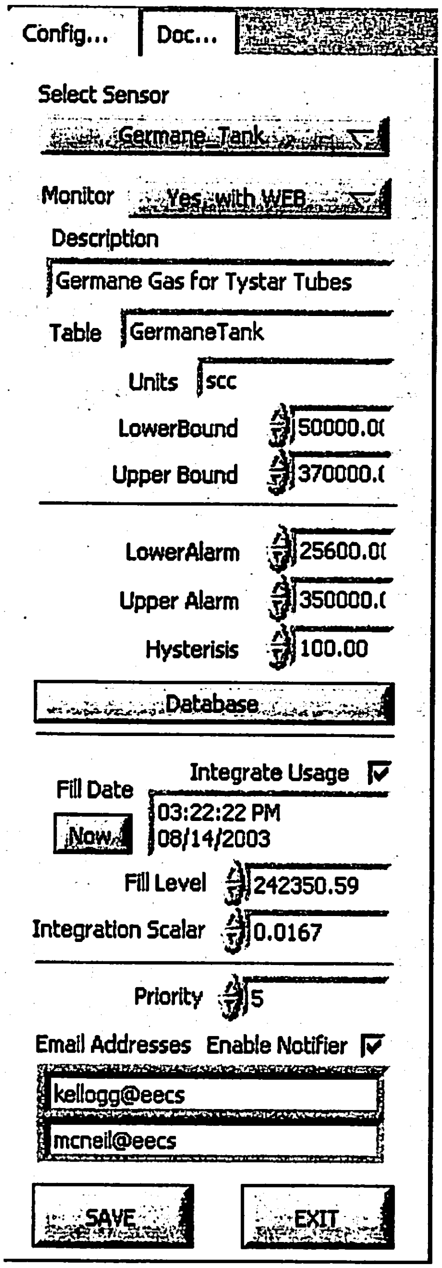


Figure 7 Sensor Configuration Panel

Select Sensor: "Drop-down" selection box allows for selecting the sensor of interest for configuration.
Description: Used to describe specific details about the sensor.
Enable Monitor: Check if data is to be collected for this sensor.
Table: This identified the table name where data is stored within the database. By default, the sensor name is used. If one changes a sensor name, the previous table of data can be used by setting this value.
Units: Physical units of the data the sensor is reporting.
Lower Bound: Lowest expected value of data.
Upper Bound: Highest expected value of data.

Lower Alarm: Lower threshold for alarm reporting.
Upper Alarm: Upper threshold for alarm reporting.
Hysterisis: This value defines the "noise" level of the data being monitored. It should normally be set to the noise specification of the sensor.
Database: Activates the UDL Connectivity file editor for database connection information (user, password, etc.)

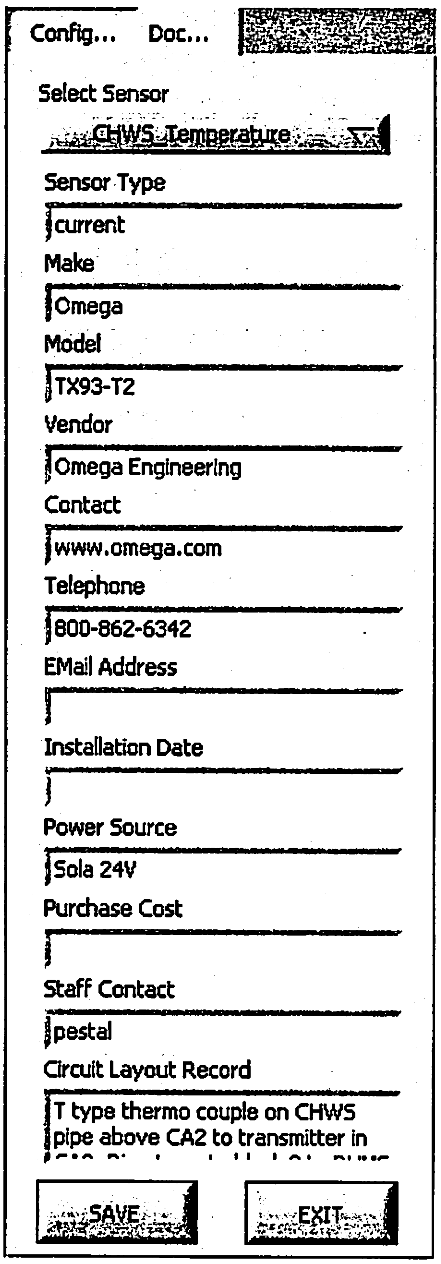
Integrate Usage: Checked for flow rate sensors
Fill Date: Last date this "Tank" was refilled.
Now: Sets the Fill Date to current time and date.
Fill Level: The fill level of this tank for the last Fill date.
Integration Scalar: Used for integrating this flow rate sensor to determine the total usage between specific times.

Priority: This value determines the priority of this sensor. Used for sorting in web page tables, etc.
Email Addresses: Recipients to receive alarm notifications.
Enable Notifier: Check to enable alarm notifications.

Save: Save the current configuration for this sensor.
Exit: Exits the program (same as File->Exit).

4.2.3 Documentation

Documentation information for each sensor is defined using the documentation sub-panel. The information defines: sensor type, manufacturer, model, contact information, and other pertinent information concerning the physical characteristics for the sensor (Figure 8).



Select Sensor: "Drop-down" selection box allows for selecting the sensor of interest for documentation

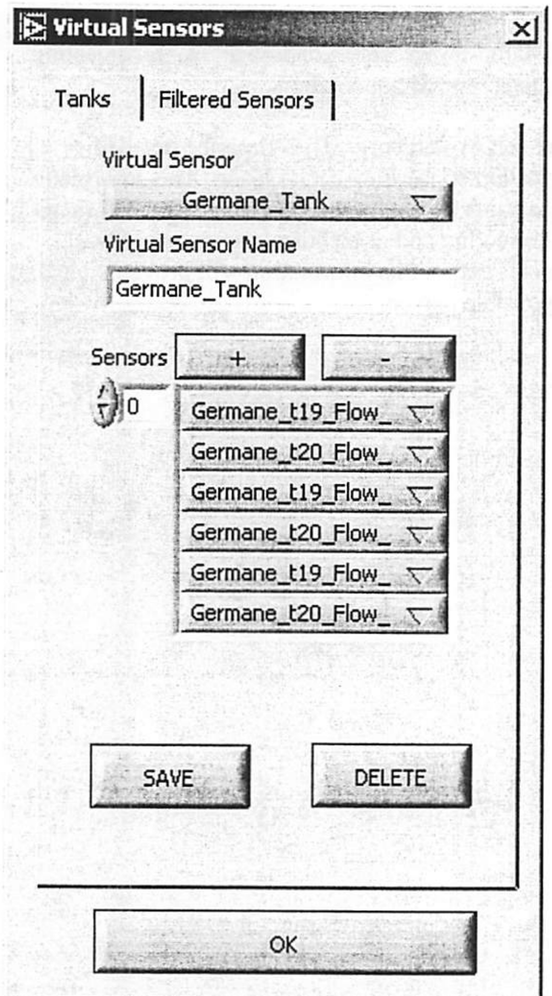
These fields are self explanatory. They describe detailed pertinent information concerning each sensor. This information enables technical staff to readily ascertain important information to aid in troubleshooting and or technical assistance. Should staff reassignments occur, this information will prove invaluable to the new responsible person.

Save: Save the current documentation for this sensor.
Exit: Exits the program (same as File->Exit).

Figure 8 Sensor Documentation Panel

4.2.4 Virtual Sensors

The Virtual Sensors panel is used to define and configure virtual sensors. Currently "Tanks" and "Filtered Sensors" are supported. Tanks are created by integrating flow-rate sensors over time and subtracting the amount of flow from the current "tank level." The "Tank" can be filled using the configuration sub-panel (Figure 9).



Virtual Sensor: "Drop-down" selection box allows for selecting the Tank of interest for configuration

Virtual Sensor Name: This will be the name of the new "Virtual Sensor"

+/-: These are used to add or subtract "real" sensors from the list of sensors associated to a specific "Tank"

Sensors: Each Drop Down box contains a list of sensors available for associated to a particular "Tank"

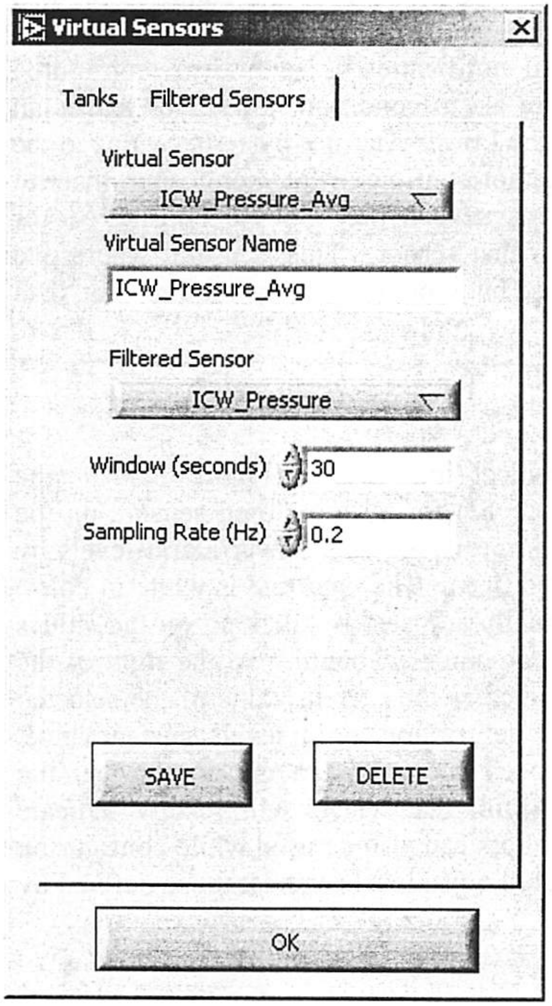
Save: Save the current configuration for this "Tank".

Delete: This will remove the "Tank" from the list of "Virtual" sensors.

OK: Exits Virtual Sensors Panel.

Figure 9 Tank Configuration Panel

Filtered Sensors currently supports only a moving average filter with a defined window size. This filter helps to eliminate minor fluctuations in sensor signal that is of no interest. Such a signal might be a building pressure that fluctuates as doors are opened and closed, but one is really interested in major changes to the pressure (Figure 10).



Virtual Sensor: "Drop-down" selection box allows for selecting the Tank of interest for configuration

Virtual Sensor Name: This will be the name of the new "Virtual Sensor"

Filtered Sensor: This defines the "real" sensor that will be filtered utilizing a moving average.

Widow: Defines the window size in seconds for the moving average filter.

Sampling Rate: Defines the specific sampling rate for this virtual sensor.

Save: Save the current configuration for this "Filtered Sensor".

Delete: This will remove the "Filtered Sensor" from the list of "Virtual" sensors.

OK: Exits Virtual Sensors Panel.

Figure 10 Filtered Sensor Configuration Panel

4.2.5 Monitoring and Recording

The server continuously monitors all sensors present in the system. That is, those that are configured through the NI MAX interface. During each pass of data sampling, a number of actions for each sensor are likely to occur. First, the current data sample is compared to the previous sample. If the absolute difference of these measurements is greater than the noise level (hysterisis configurational parameter) and monitoring is enabled for that sensor, then a new data record is generated in the data table of the database for that sensor. Next, if "Enable Notification" is enabled, and the current data is outside the Lower to Upper Alarm settings, then an email notification is sent to all configured recipients if the sensor was not previously in an alarm condition. A sensor leaves an alarm condition by having the data fall between the Lower Alarm + hysterisis value to the Upper Alarm - hysterisis value. If the sensor is not configured for monitoring, then no data is stored for that sensor. Even though a sensor may not be configured for monitoring, alarm notifications still function for that sensor. This is useful where one would want to be notified of alarm conditions, but not necessarily require that data history be available.

4.2.6 Real-Time Graphical Analysis

The server and clients display real-time graphical information (Figure 11) for the currently selected sensor and tabular data (Figure 12) for all monitored sensors on the front panel "Real-Time" tab. The graph displays the Upper and Lower alarm levels by red lines and the hysterisis levels by green dotted lines. The data line is white in color. The upper and lower y-scale limits can be manually adjusted by clicking on the values with the computer mouse and editing with the computer keyboard. To the right of the data graph is a "Bar" graph that also visually indicates the current value of the selected sensor. The tabular data consists of Sensor name, description, alarm levels, current value, and alarm condition. Only the sensors that have been selected for monitoring through the configuration tab are displayed in the table. Both data views will readily indicate whether there is an alarm condition. These indicators can also be used while configuring a new sensor to verify the correctness of configured noise levels against those defined by the specific sensor specifications.

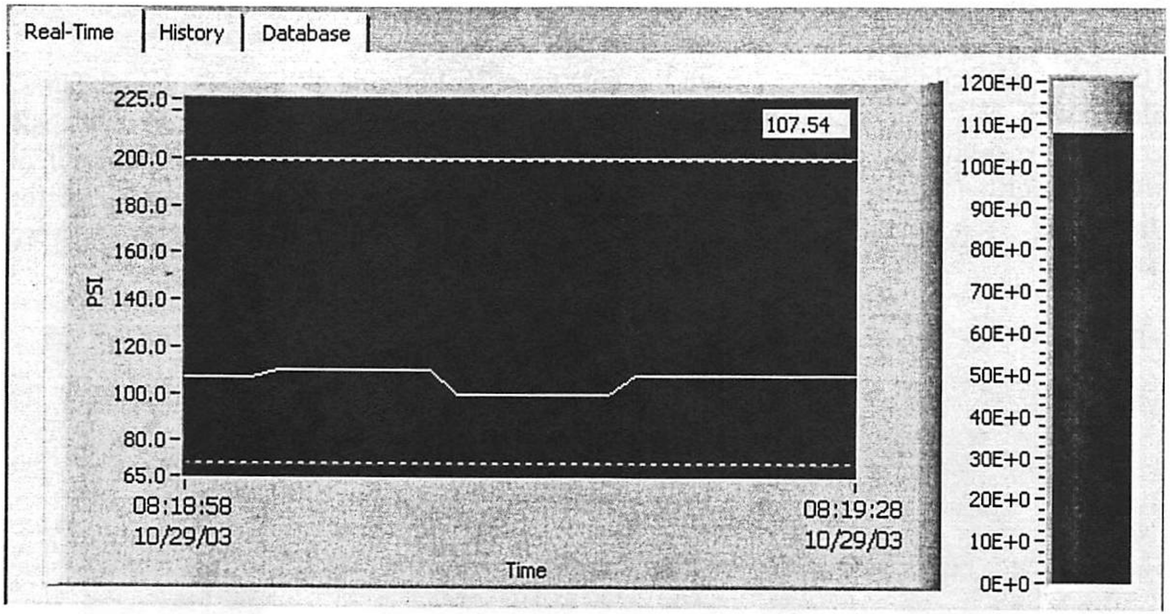


Figure 11 Real Time Display

Sensor	Description	Low Alarm	Upper Alarm	Value	Alarm?
DI_Makeup_Flow_Rate	DI water makeup flow r.	-1.00 gpm	5.00 gpm	0.94 gpm	No
N2_Flow_Rate	Nitrogen flow rate	200.00 slpm	1400.00 slpm	964.74 slpm	No
HAZMAT_Toxic_Gas_Ala	HAZMAT toxic gas dete	-1.00 on/off	0.50 on/off	0.00 on/off	No
Cooling_Tower	Cooling tower pump	0.50 on/off	2.00 on/off	1.00 on/off	No
HF_70	Exhaust hood fan 70 (1	0.50 on/off	2.00 on/off	1.00 on/off	No
HF_83	Exhaust hood fan 83 (4	0.50 on/off	2.00 on/off	1.00 on/off	No
HF_84	Exhaust hood fan 84 (G	0.50 on/off	2.00 on/off	1.00 on/off	No
Makeup_Air_Temperature	Supply air temperature	55.00 Deg. F	70.00 Deg. F	65.23 Deg. F	No
Germane_t19_Flow_Rate	Tystar19 germane flow	-3.00 sccm	3.00 sccm	-0.74 sccm	No
Disilane_t19_Flow_Rate	Tystar19 disilane flow ra	-3.00 sccm	3.00 sccm	-0.06 sccm	No
DI_Resistivity	Resistivity of DI water	18.00 Mohm	19.00 Mohm	18.31 Mohm	No
Drain_Vacuum	Vacuum pressure for lat	10.00 inches Hg	27.00 inches Hg	23.83 inches Hg	No
N2_Pressure	Nitrogen pressure	75.00 psi	95.00 psi	91.75 psi	No
O2_Pressure	Lab oxygen pressure	10.00 psi	40.00 psi	30.59 psi	No

Figure 12 Real Time Data in Tabular Format

4.2.7 Historical Graphical Analysis

Historical data can be retrieved from the data store by entering a "Start Date and Time" along with an "End Date and Time", and then select "Get Data." Along with the graphical display, the data is also displayed in tabular format (Figure 13). The client software further allows for exporting the data to a tab separated values format file for displaying in a spreadsheet program or for importing into other analysis software packages.

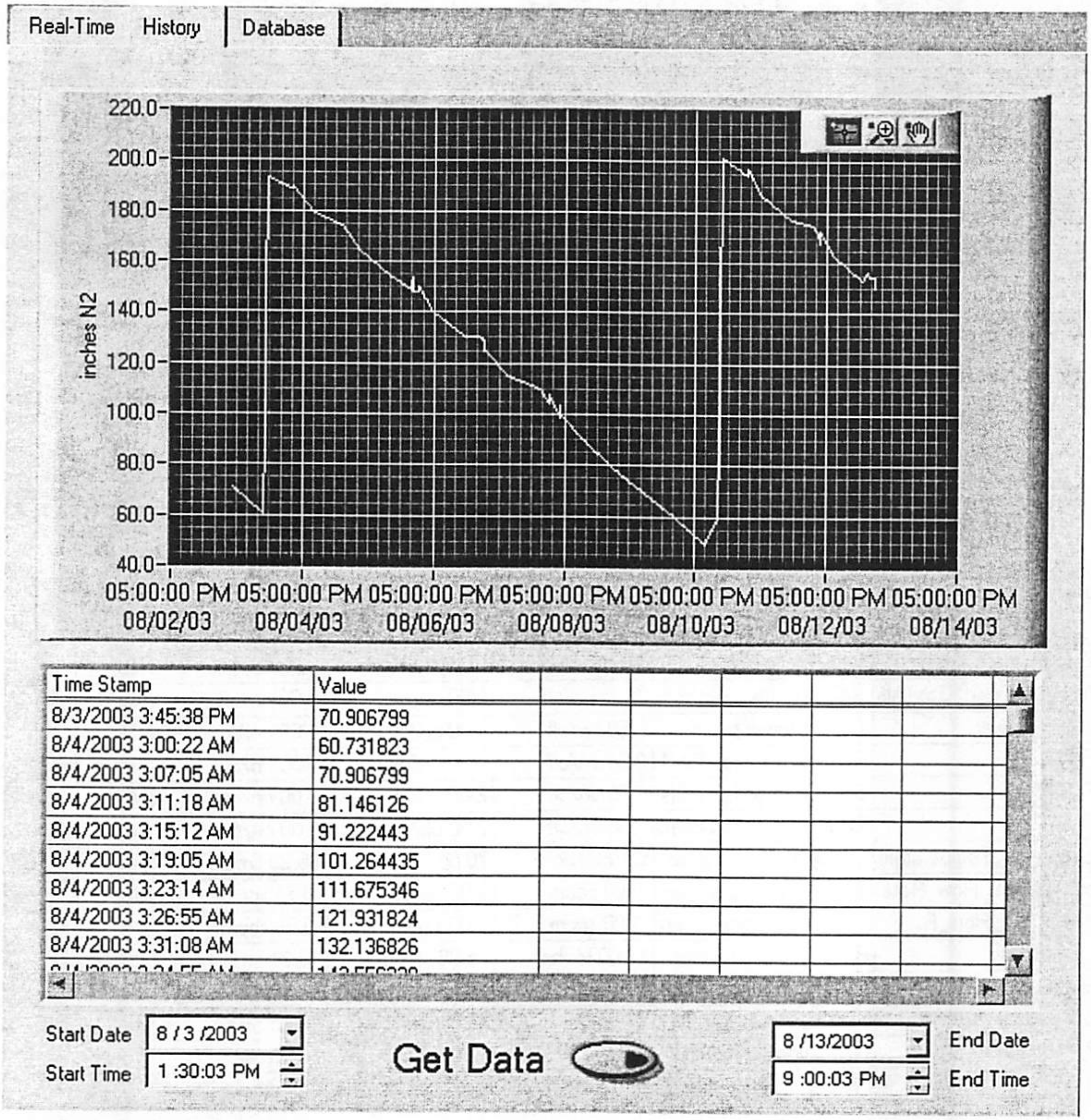


Figure 13 Historical Graphic and Data Display

4.2.8 Web Page Auto-Generation

For viewing on the web, 30-day historical graphs are automatically generated every hour for each sensor being monitored. Figure 14 shows two representative samples. Upper and lower alarm limits are shown in red on the screen. Y-Scale Max and Min values are the Upper and Lower Bounds respectively. The sensor name is given in the title along with the most recent value and time at graph generation. The sensor description is given under the time axis. All these parameters are configured from the configuration section of the RUMS server interface.

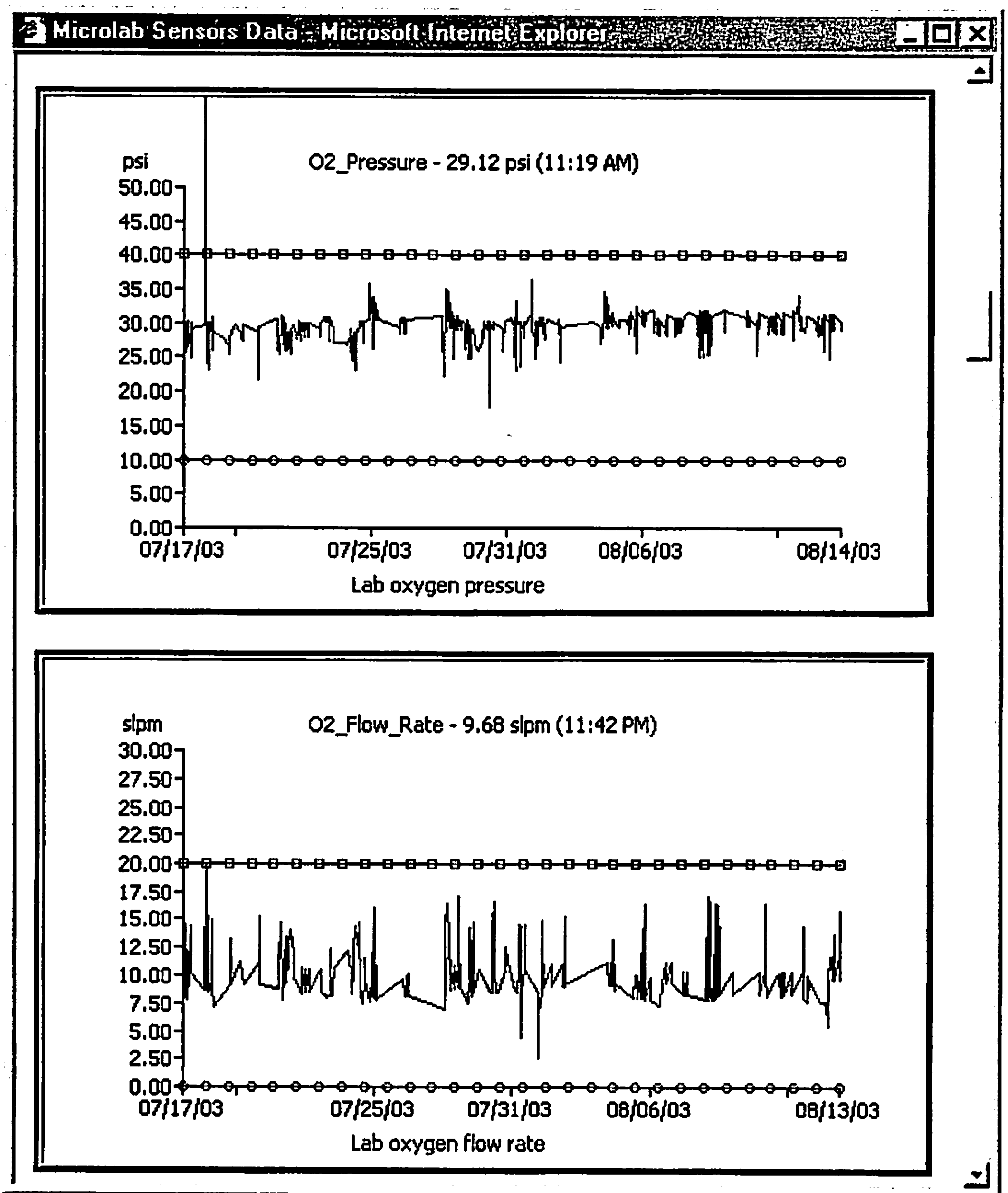


Figure 14 Web Page Graphs

4.2.9 Database Schema

The database schema consists of tables for storing configuration parameters, sensor documentation, virtual sensors to tank mappings, data table name references, current sensor values, and data tables for each sensor to store the actual data. The configuration table stores the alarm settings, email notifies, monitoring status, upper/lower bounds and the rest of the configurable information defined in the configuration pane. The documentation table likewise stores the relevant field parameters as defined in the documentation tab. The tanksensors table stores the TankName and sensor name for each sensor "attached" to the tank. The tablenames table maps the sensor name to the table name where the raw data for that sensor is stored. The current values table stores the most recent value for each sensor along with a time stamp. The germane tank table in Figure 15 below is representative of a typical data table for a single sensor, whether a regular or virtual sensor.

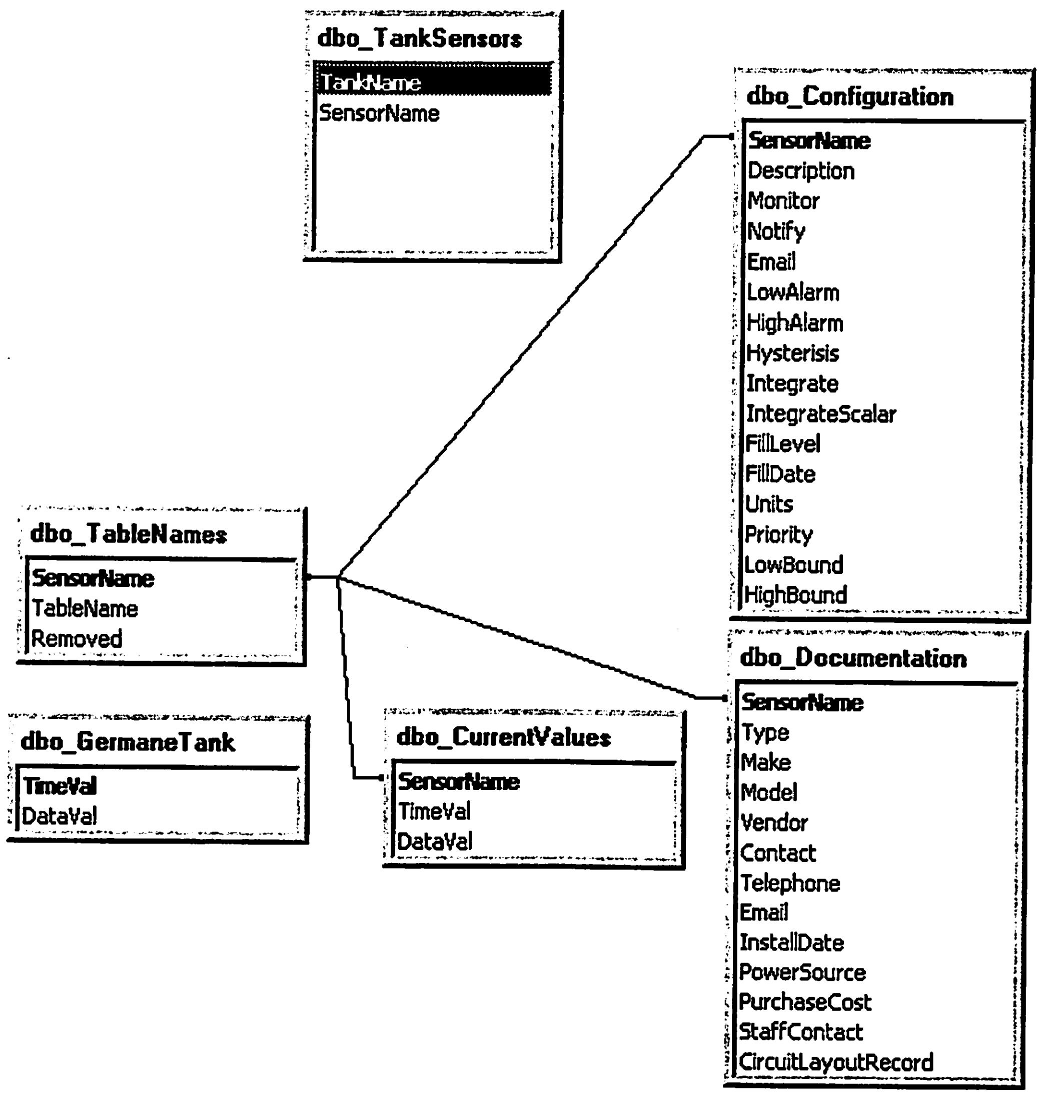


Figure 15 Database Schema

4.3 Autonomous Web Interface

4.3.1 Database Backend

Web technology has now become a common interface for business or organizations to share/exchange information. The emerging and popular use of the web also promotes rapid development in this area. In this project (RUMS), we adopt the latest .NET technology to develop a web based RUMS client interface in C# (the new programming language from Microsoft). This web based client interface provides ease of use, a standard user interface, and platform independence with no client software distribution required.

Figure 16 shows a basic design model for the web based RUMS client interface.

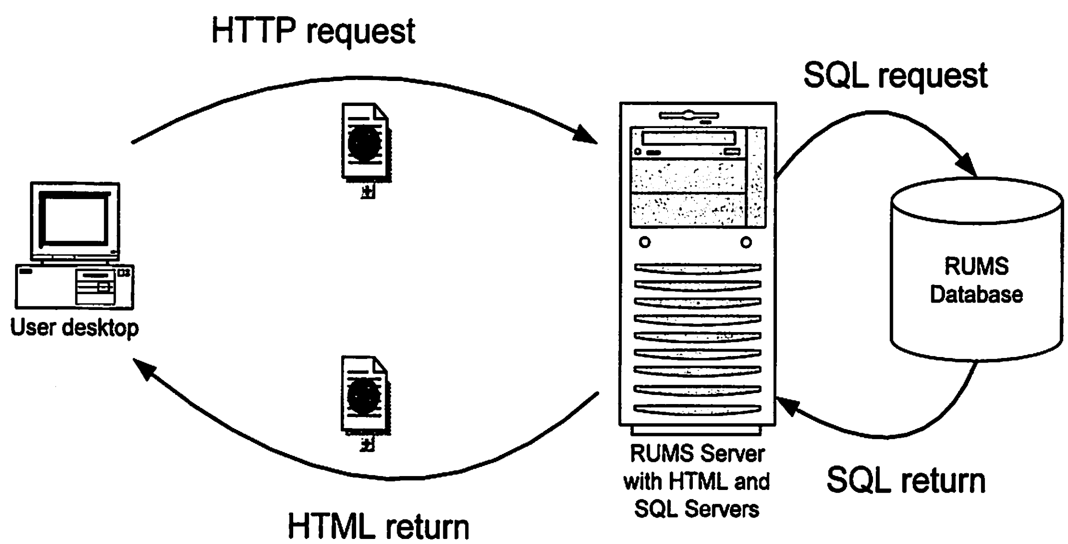
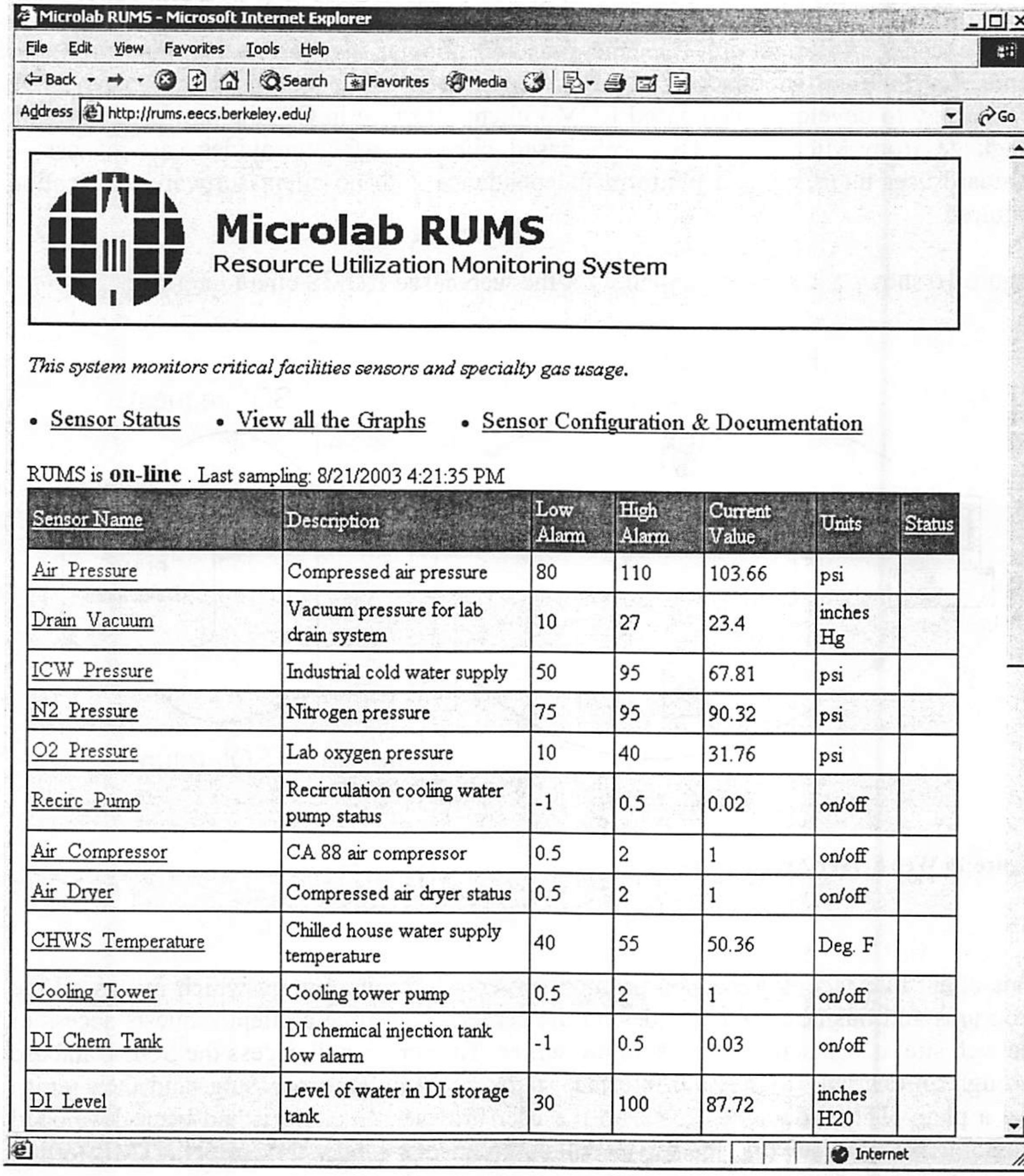


Figure 16 Web Server Architecture

This client interface is generated through server-side programming which means all the programs and business logic resides on the server. Each time a client requests access to the web site, it will send a request to the server. The server will access the SQL Database through an OLEDB connection interface, perform complex processing, and then return just a plain HTML document back to the user browser. Because it had been developed upon the .NET framework, the server will automatically render the correct HTML format to the client based on the type of browser used by the client.

4.3.2 Sensor Status Page


Microlab RUMS - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Address http://rums.eecs.berkeley.edu/

Microlab RUMS
Resource Utilization Monitoring System

This system monitors critical facilities sensors and specialty gas usage.

- Sensor Status
- View all the Graphs
- Sensor Configuration & Documentation

RUMS is **on-line**. Last sampling: 8/21/2003 4:21:35 PM

Sensor Name	Description	Low Alarm	High Alarm	Current Value	Units	Status
Air Pressure	Compressed air pressure	80	110	103.66	psi	
Drain Vacuum	Vacuum pressure for lab drain system	10	27	23.4	inches Hg	
ICW Pressure	Industrial cold water supply	50	95	67.81	psi	
N2 Pressure	Nitrogen pressure	75	95	90.32	psi	
O2 Pressure	Lab oxygen pressure	10	40	31.76	psi	
Recirc Pump	Recirculation cooling water pump status	-1	0.5	0.02	on/off	
Air Compressor	CA 88 air compressor	0.5	2	1	on/off	
Air Dryer	Compressed air dryer status	0.5	2	1	on/off	
CHWS Temperature	Chilled house water supply temperature	40	55	50.36	Deg. F	
Cooling Tower	Cooling tower pump	0.5	2	1	on/off	
DI Chem Tank	DI chemical injection tank low alarm	-1	0.5	0.03	on/off	
DI Level	Level of water in DI storage tank	30	100	87.72	inches H20	

Internet

Figure 17 Sensor Status Web Page

Figure 17 shows a real time sensor status page, which is also the front page of the RUMS web based client program. Every time this page is requested, it goes to the database and collects the current value of the sensor data and its associate configuration information. The information is then displayed in data grid format. The status of sensor in an alarm state will show up as a flashing red light. Each sensor name is also a hyperlink to its data graph generated by the LabVIEW program.

This page also provides the real status of the RUMS system; it will show whether RUMS is on-line or off-line as well as its last sampling date and time.

The information collection process is accomplished through a stored procedure (rums_web) running on the RUMS SQL Database Server. This stored procedure runs every time the web page is requested to provide the required real time information based on the most recent sensor values currently stored in the database. This stored procedure creates a temporary table that stores all the information necessary for displaying the current sensor status information. Next the asp.net page displays the proper information on the web based on the records from this table. It also has the sorting capability to allow users to organize the data based on specific fields.

4.3.3 Sensor Web Graphs

Another asp.net page dynamically links all the sensor data graphs to one page based on the sensor's configuration in the RUMS database. The sensor graph, such as shown in Figure 14, will be displayed for each sensor that is configured to "display on the web."

4.3.4 Sensor Web Configuration

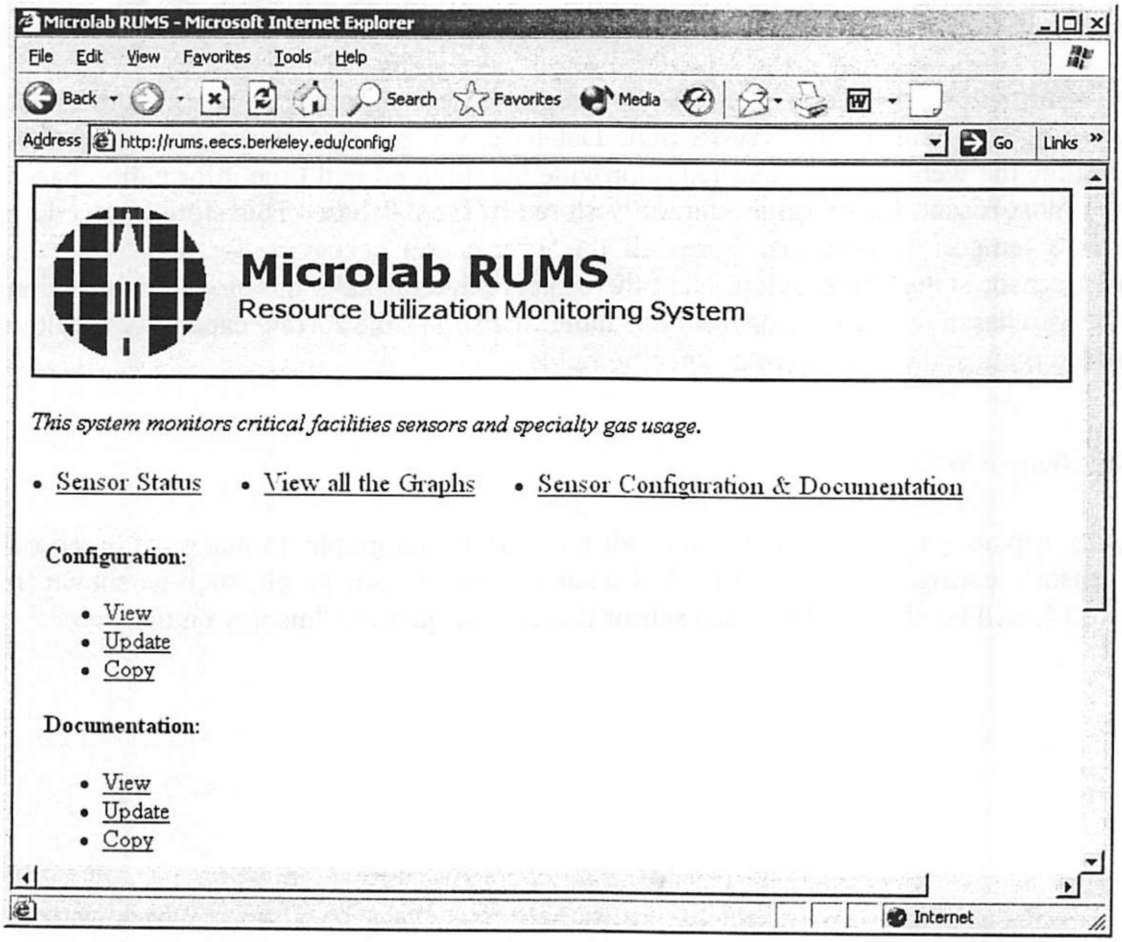


Figure 18 Sensor Web Configuration Main Page

The Sensor Configuration and Documentation page (Figure 18) provides three major capabilities. View, Update, and Copy. The "View" page provides a way of viewing the sensor configuration or documentation data only. The information cannot be modified. This page was designed to give users a tool for retrieving information only.

The "Update" page (Figure 19) provides the capability to modify the configuration and documentation records in the database. Once the "Update" button is clicked, a series of actions will take place in the background. First, the database record will be updated based on the information that the user provided in the HTML form. Next, a database update trigger will be initiated which will result in the execution of a "client.exe" program. This program is built in LabVIEW environment, and when it runs, it will initiate a socket connection to the RUMS server and send an init signal (covered in Section 4.4) to the RUMS server before exiting. Once the RUMS server receives this init signal, it will re-initialize itself by reloading all the configuration and documentation information from the database into the program. After the processes are completed, the user will receive a "Sensor configuration/documentation has been successfully updated" message. A notification email message will also be sent to all RUMS administrative staff.

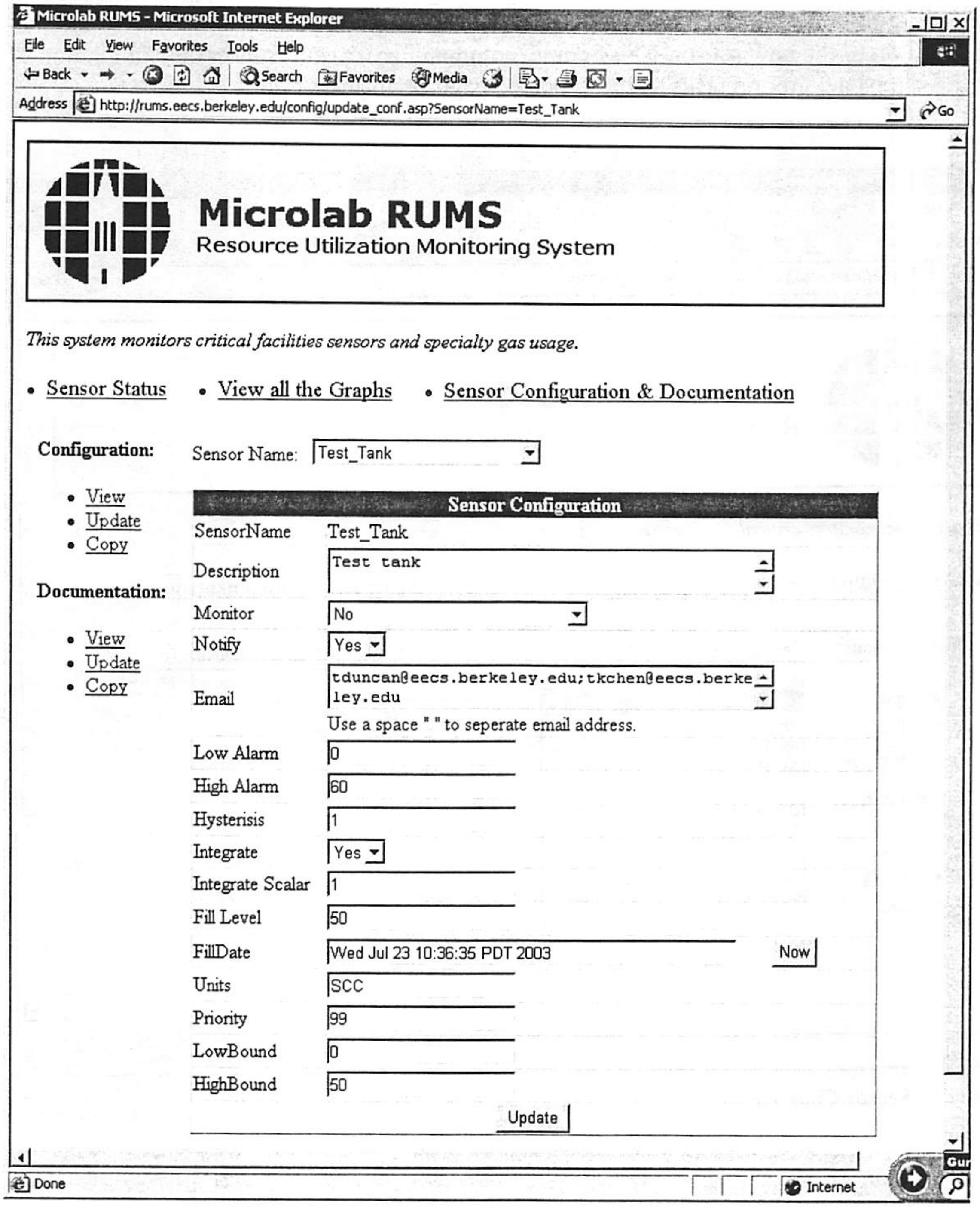


Figure 19 Sensor Web Configuration Detail

The "Copy" page (Figure 20) provides staff a convenient way of managing sensor information. Instead of creating all the information from scratch for a new sensor, one can go to this page and copy any existing sensor configuration and/or documentation information to the new sensor. This "copy" command will also result in the completion of all the same background tasks as the "update" page performs.

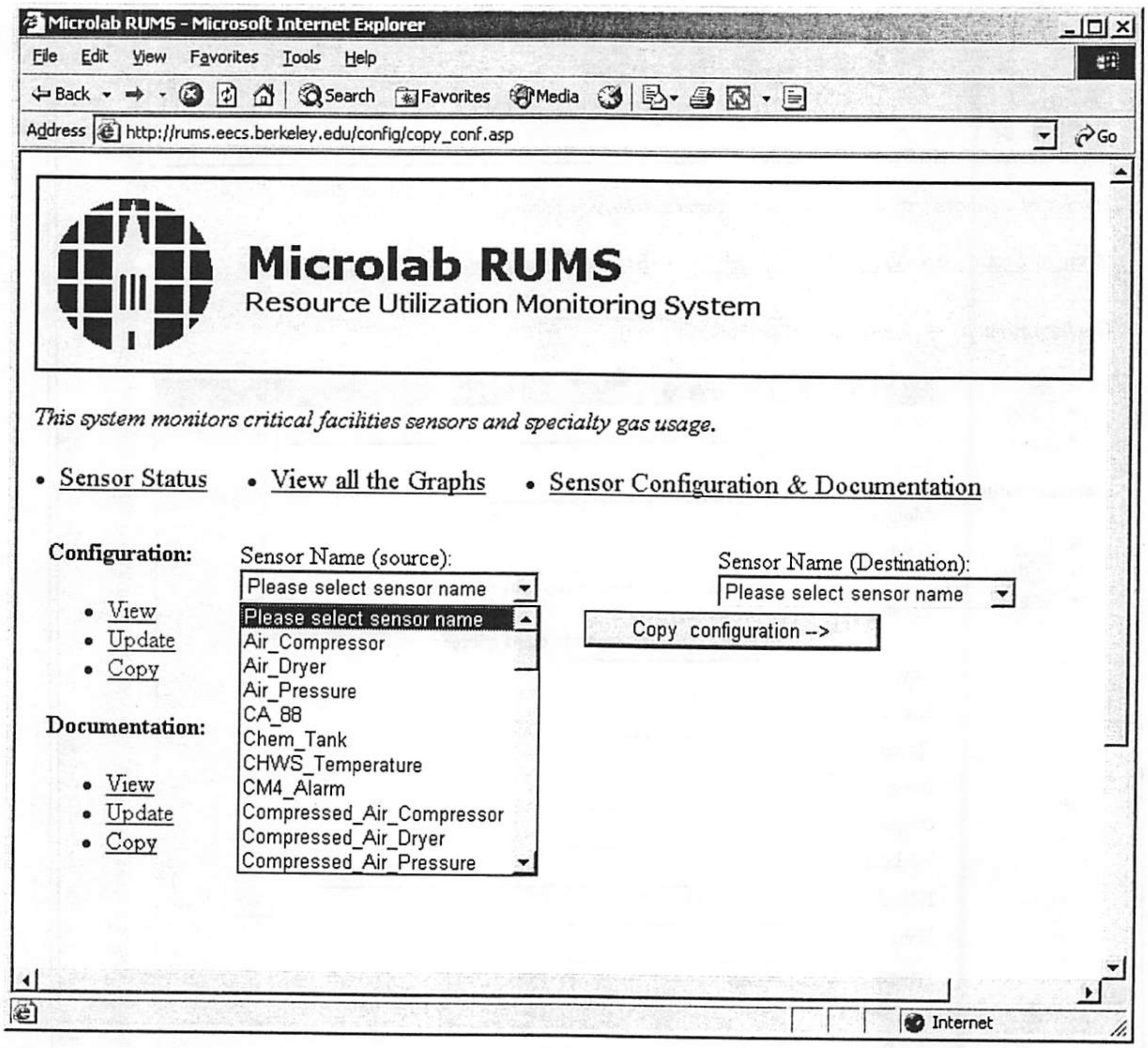


Figure 20 Sensor Copy Page

4.4 RUMS Resource Server

The RUMS resource server acts as a communication gateway to the RUMS system for querying resource usage (Figure 21.) Network sockets are used to connect to the RUMS resource server and communicate using a simple plain-text command and response protocol. Sensor resource usage, such as that obtained from integrating flow rate sensors, may be retrieved for any specified time interval.

This resource server module is compiled into the RUMS Server application with multi-threaded support. This allows for multiple concurrent connections to the RUMS resource server. It also properly handles thread safety as well as timeouts. The RUMS resource server is an integral component of the RUMS system.

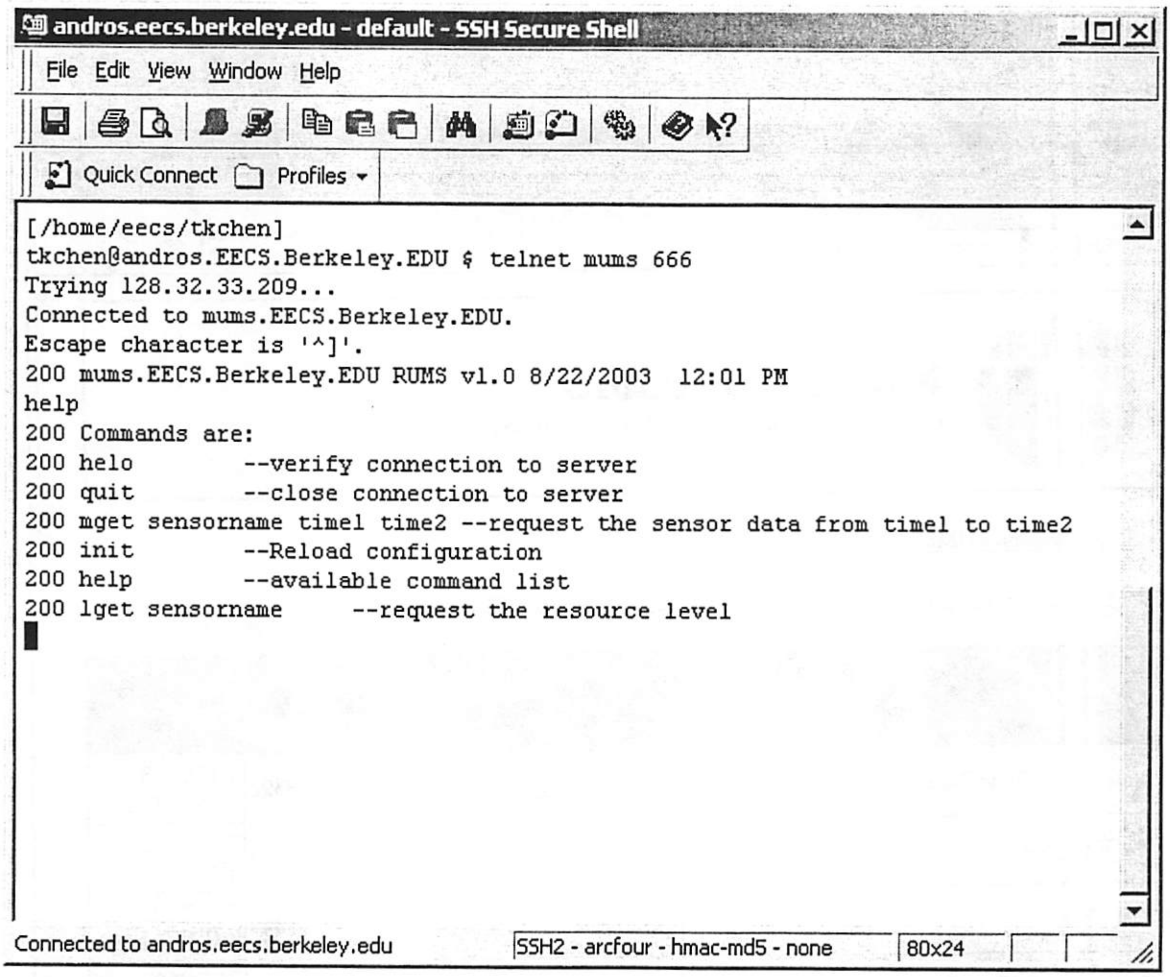


Figure 21 Resource Usage Server (using a telnet client connection)

There are six different four-letter commands recognized by the server.

1. helo – verify connection to server
2. quit - close connection to server
3. mget sensorname time1 time2 – request the sensor usage data from time1 to time2
4. init – reload the configuration data from the database
5. help – available command list
6. lget sensorname – request the resource level (from a tank)

4.5 RUMS Administrative Web Site

Since the database stores data for all the sensors, database maintenance is of prime importance. The RUMS system generates a large number of records; thus it is essential to keep track of the health of the database. A RUMS admin web site was developed to give administrative staff an easy and effective way of monitoring and tracking database performance.

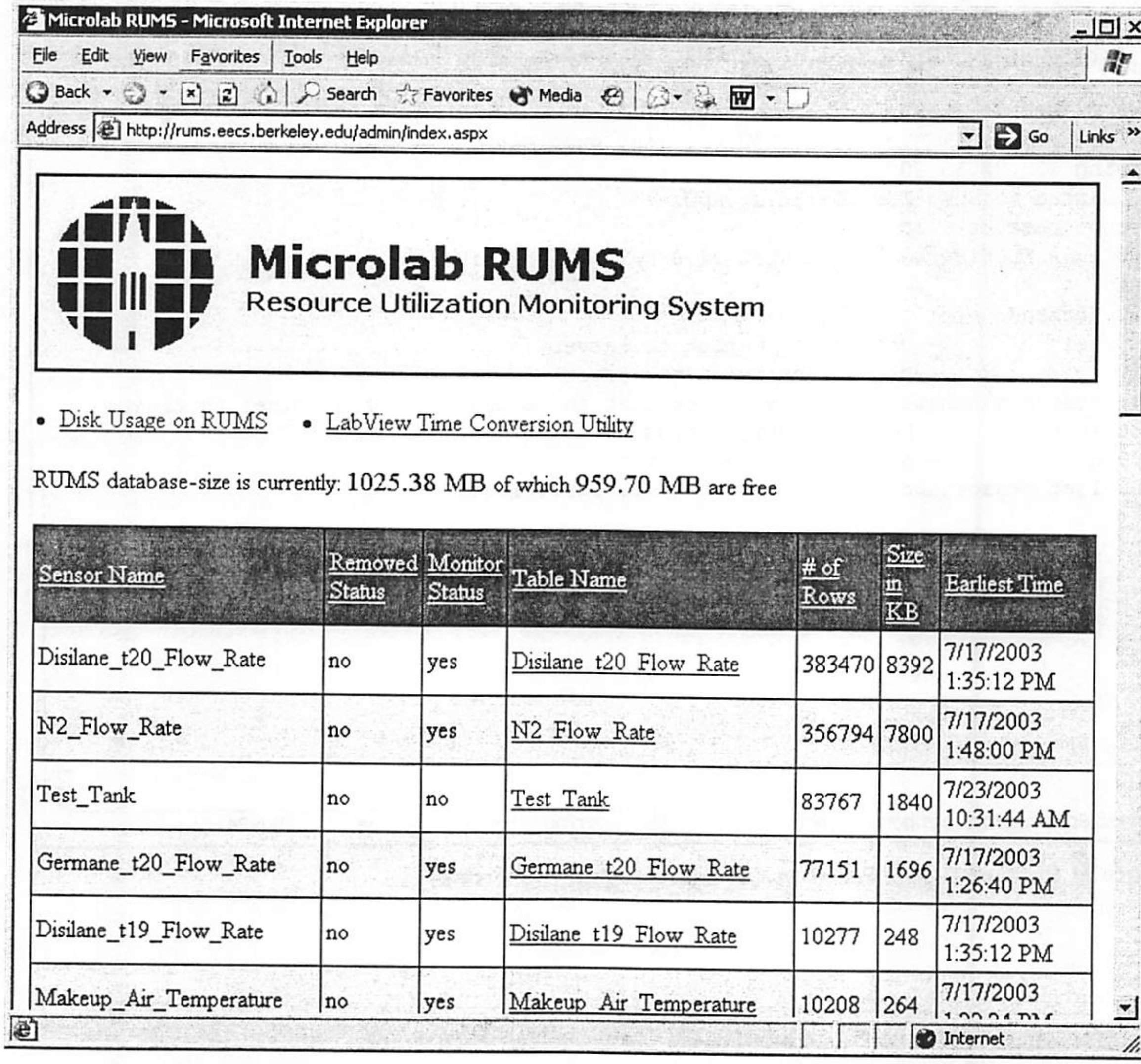


Figure 22 Database Administrative Web Site

4.5.1 Disk Space Availability

From the main page (Figure 22), the current database size and unallocated space is immediately available. Also displayed are the metrics for each sensor. Information available for each sensor includes the name, its associated data table name, status of the data table, the number of rows in the data table, size of the data table, and the earliest time record.

4.5.2 Trend of Number of Records

Available for each sensor is a trend of the number of records stored in each data table. This graph may be viewed by clicking on the table name link. To provide the trend of the table records, an ADMIN database was created with identical table names as the RUMS tables. A stored procedure, (rums_admin) is responsible for populating all the tables at a weekly interval. This tracks the number of records in each sensor table. Figure 23 illustrates the record growth trend for the ICW_Pressure sensor.

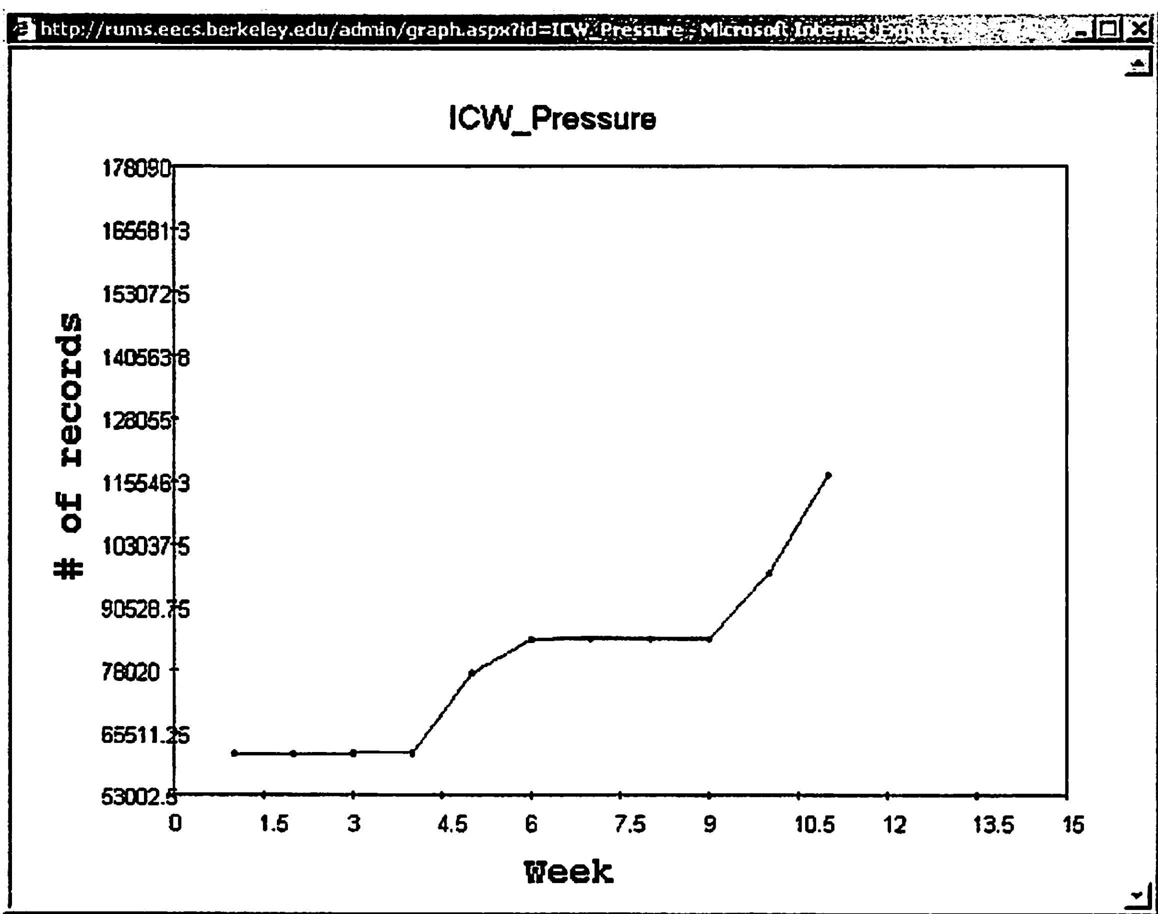


Figure 23 Database Trend Graph

4.5.3 Disk Usage

By clicking the "Disk Usage on RUMS" link, the total disk usage of the RUMS database is displayed; it is presented in both graphical and numeric format (Figure 24).

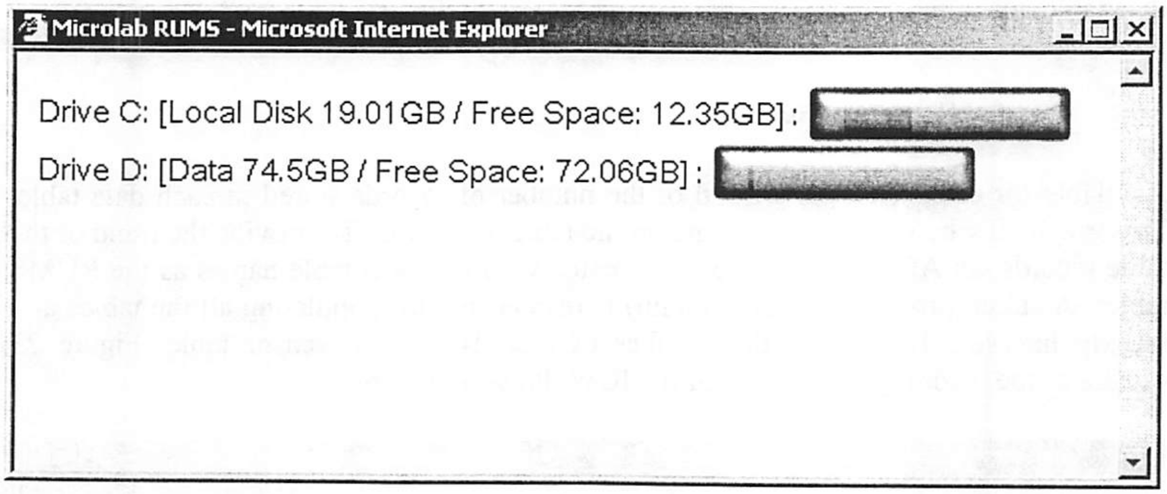


Figure 24 Database Disk Usage

4.5.4 LabVIEW Time Conversion Utility

By clicking the "LabVIEW conversion utility" link, a utility is available to convert a local time format into LabVIEW time (Figure 25). This is required for the archive utility (covered in the following paragraph.)

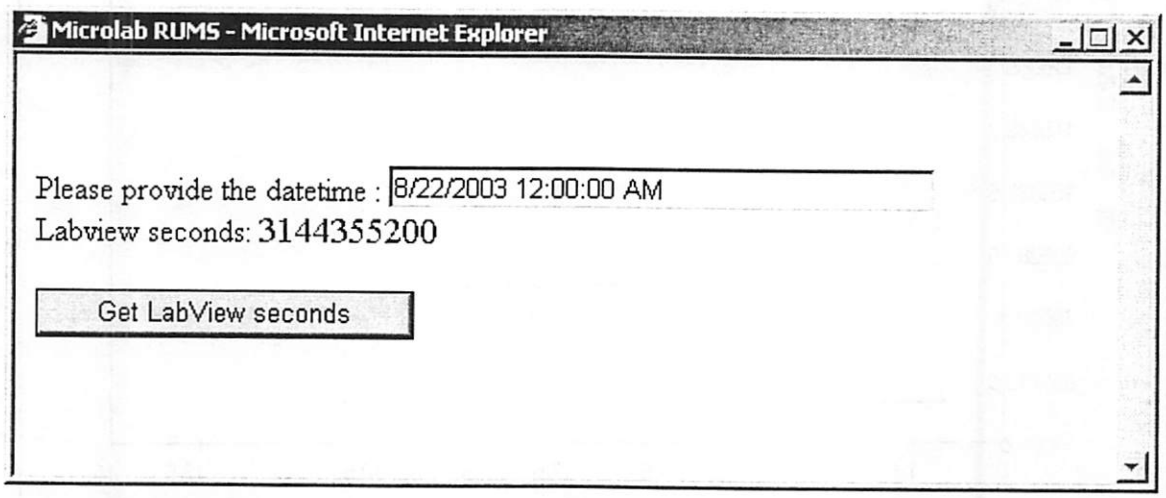


Figure 25 Database Time Converter

4.5.5 Archiving

The Archive utility program takes a parameter for the cutoff date and time (in LabVIEW format.) It uses this parameter to identify every record stored in every sensor data table in the RUMS database that precedes the cut-off time. These records are copied to the ARCHIVE database with the same analogous table name. (It will create a table with the same name if it does not already exist.) It then deletes the obsolete records from the RUMS main database tables. All these action are done in atomic transaction mode to insure data integrity. If an error occurs during the processing of an individual table, the transactions for that table are rolled back, thus insuring the required data integrity.

4.6 RUMS Watchdog Program

RUMS is remotely monitored by a watchdog program written in the Perl programming language (chk_rums.pl) to insure problems with the system can be quickly detected and resolved. It performs three tests to determine the health of RUMS. The first test checks network connectivity. If successful, a handshake connection to the RUMS resource server will be attempted, followed by a specific resource query. Mail is sent to system administrators when a problem is detected. The watchdog program incorporates latching and hysterisis mechanisms to prevent an unwanted flood of mail messages. The watchdog program has a configurable heartbeat of 10 minutes. The program is run as a daemon on a UNIX system.

5 SUMMARY

RUMS is a versatile and powerful system for monitoring critical utilities and resources in the Microfabrication Laboratory. The technology deployed for RUMS utilizes industry standard technologies (LabVIEW, C#, and SQL Server) to insure the system can modified and easily upgraded as new needs arise. RUMS is a stand-alone system to insure maximum reliability through network independence, yet provides network connectivity and real-time access to other Microfabrication Laboratory systems. RUMS is currently collecting data from 28 critical sensors.

REFERENCES

LabVIEW Users Manual: National Instruments Corporation, July, 2000.

Learning with LabVIEW, Robert H. Bishop, Prentice Hall, ISBN: 0-13-032559-7

LabVIEW Hands-ON Seminar Manual: An Introductory Look at Graphical Development. Austin, TX: National Instruments Corporation, 2002.

Data Acquisition and Signal Conditioning Course Manual. Austin, TX: National Instruments Corporation, 2002.

SCB-100 100-Pin Shielded Connector Block Installation Guide. Austin, TX: National Instruments Corporation, 1995.

DAQ. PCI E Series User Manual. Multifunction I/O Devices for PCI Bus Computers. Austin, TX: National Instruments Corporation, 2002.

BIOGRAPHIES

Tim Duncan received his B.A. degree in Physics at the University of California, Berkeley, CA. in 1998, and an A.S. degree in Electronics Technology at the College of the Redwoods, Eureka, CA. in 1994. Upon graduation in 1998, he worked as a Programmer/Analyst for a graduate research group and the Microfabrication Laboratory at U.C. Berkeley (Microlab) providing custom application development to support both the infrastructure of the many varied semiconductor processing equipment present in the lab and to support researchers with custom application development for use in specialized research activities. After leaving for a period of two years to work in a semiconductor equipment novel sensor manufacturing start-up (OnWafer Technologies, Inc.) as principal software engineer, he returned to work for the Microlab full time as a Programmer/Analyst to develop and implement a new database driven enterprise fabrication facility management infrastructure system based on a newly developed XML based framework. In his spare time, he enjoys bike-riding and scuba-diving in warm-climate areas.

T. K. Chen received the B.S. degree in mechanical engineering from Chung-Yuan Christian University, Taiwan, in 1992 and an M.S. degree in computer science from Oklahoma State University, Oklahoma in 1998. He was a Member of Technical Staff in the IT department at Lucent Technologies from 1998 to 2000. He then joined Intel Corporation as a Software Engineer at Portland, Oregon from 2000 to 2002. During this time, he worked at Intel's state of the art development fabrication manufacturing environment responsible for computer automation specializing in database management. Currently, he is working in the Microlab at the University of California at Berkeley as a programmer analyst. He is responsible for developing specialized software applications for the lab as well as managing the computing infrastructure. His research interests include information storage and retrieval, web and database integration, etc. He also enjoys playing volleyball, swimming and hiking.

Danny Pestal received a B.S. degree in Bioengineering from the University of California at Berkeley in December 2001. While a student, he worked at the U.C. Berkeley Microfabrication Laboratory as a Lab Assistant. After graduation, Danny was hired by the lab to work fulltime as a Development Engineer. He is responsible for the up keep of thin-film testing and inspection tools along with management of the hardware side of the lab's equipment control system and the resource utilization management system.

Todd Merport received an A.S degree in Electronic Engineering Technology from Heald College, San Francisco, CA in 1981. He was an Engineering Technical Associate at Dalmo Victor (part of Bell Aerospace) from 1982 to 1984 where he was involved in the development of power systems for radar warning receivers. From 1984 to 1997 he was a Development Engineer for the Civil and Environmental Engineering Dept. at the University of California, Berkeley. He developed data acquisition and control systems and software for structural research projects including retrofit tests for the Bay and Golden Gate bridges. From 1998 to 2000 he worked for Agilent Technologies developing manufacturing test software for Network Analyzers. He is currently the Computer Systems Manager for the Microfabrication Laboratory at U.C. Berkeley.

APPENDIX Software Flow Diagrams

The significant software flow diagrams for the RUMS system are shown below. The first is the main RUMS server. This is followed by the RUMS resource server (to query gas usage). Finally, a diagram for the RUMS resource client is shown.

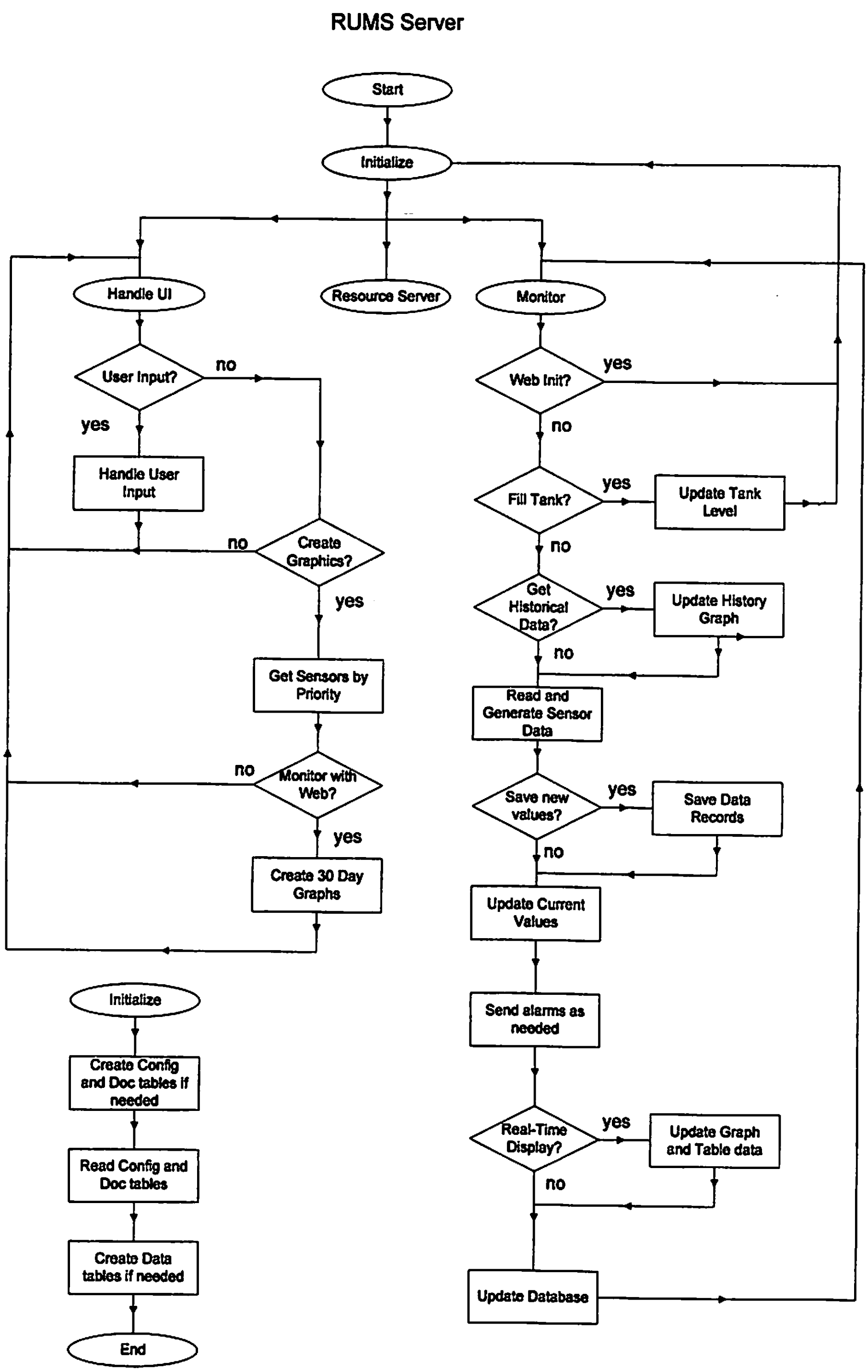


Figure 26 RUMS Server Flow Diagram

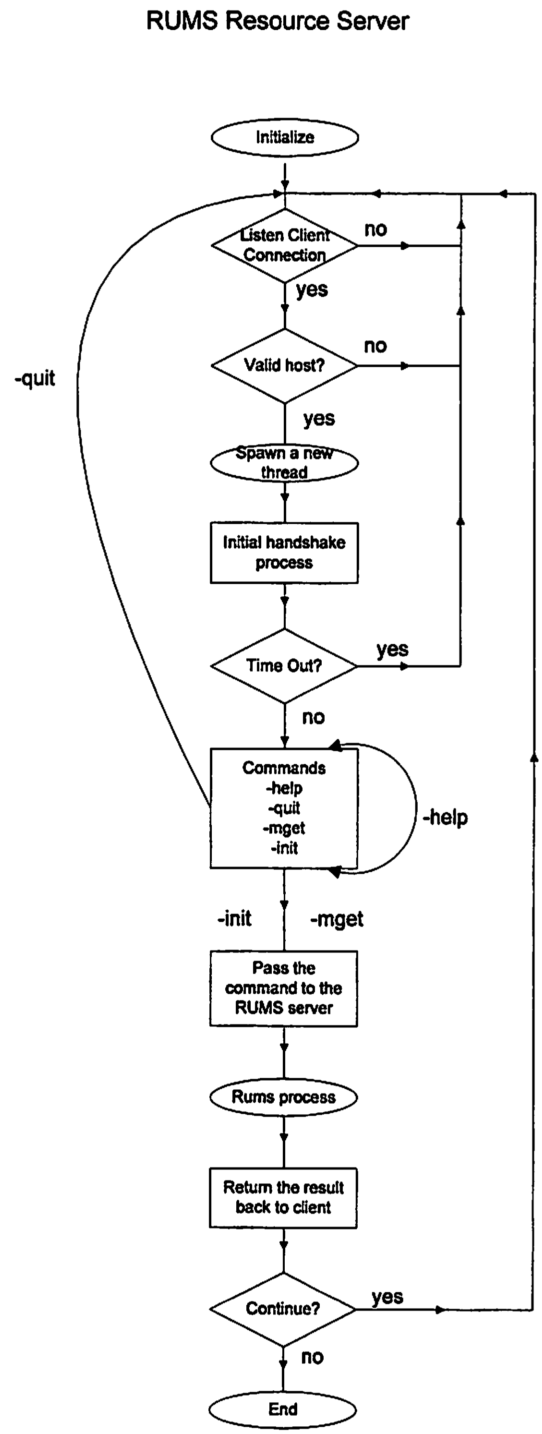


Figure 27 Resource Server Flow Diagram

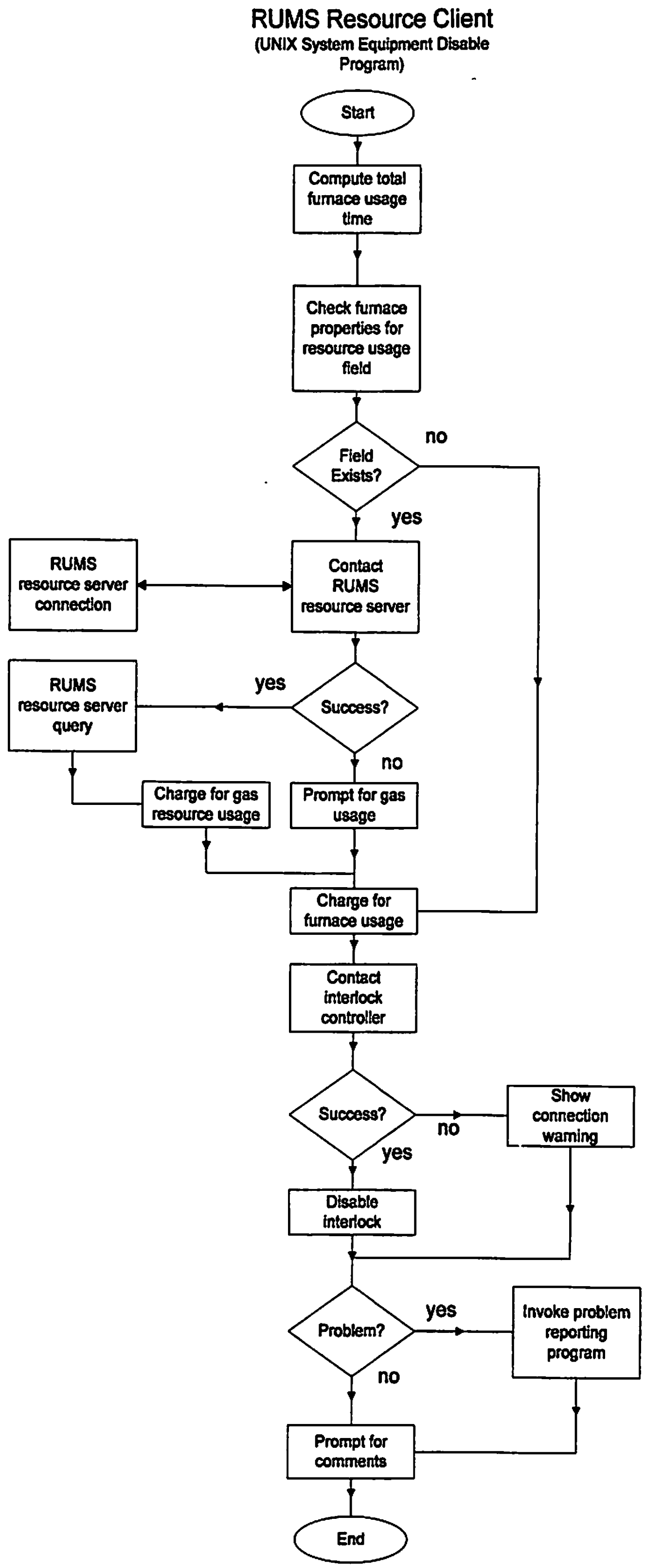


Figure 28 Resource Client Flow Diagram