MEMBRANE PROCESS OPTIMIZATION TECHNOLOGY

PerLorica Inc.
Grass Valley, California

Assistance Agreement 02-FC-81-0833

Desalination and Water Purification Research and Development Program Report No. 100

August 2003

United States Department of the Interior
Bureau of Reclamation
Denver Office
Technical Services Center
Environmental Services Division
Water Treatment Engineering and Research Group

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave Blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	August 2003	Final

4. TITLE AND SUBTITLE

Membrane Process Optimization Technology

5. FUNDING NUMBERS

Assistance Agreement
02-FC-81-0833

6. AUTHOR(S)

Thomas D. Wolfe

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

PerLorica, Inc.
10565 Brunswick Road, Suite 9
Grass Valley, CA 95945

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Bureau of Reclamation
Denver Federal Center
P.O. Box 25007
Denver, CO 80225-0007

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

Desalination and Water Purification Research and Development Report No. 100

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Available from the National Technical Information Service,
Operations Division,
5285 Port Royal Road, Springfield, Virginia 22161

12b. DISTRIBUTION CODE

13. ABSTRACT ***(Maximum 200 words)***

This project developed and refined methods for predicting the time to the next required maintenance for membrane systems based on a number of different criteria. A "Days To Next Cleaning" (DTNC) is computed from historical data based on statistically significant changes in key operating parameters which are derived from large numbers of raw and normalized operating values. Operating data from a 3 year period, taken approximately every 10 minutes, for a 1 MGD reverse osmosis system and a 1 MGD nanofiltration system, were used to validate and test the algorithms, which have proven to give useful predictive results. To test and demonstrate the system the methods and algorithms were incorporated into a web based commercial product called WaterEye® which allowed the results for the test system as well as other systems to be continuously displayed via the web.

14. SUBJECT TERMS

Monitor, membranes, reverse osmosis, nanofiltration, normalization, prediction, membrane cleaning, membrane treatment, internet, database, web, drinking water, flux, differential pressure, scaling, fouling, bio-fouling

15. NUMBER OF PAGES

54

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UL	UL	UL	UL

MEMBRANE PROCESS OPTIMIZATION TECHNOLOGY

Thomas D. Wolfe

PerLorica Inc.
Grass Valley, California

Assistance Agreement 02-FC-81-0833

Desalination and Water Purification Research and Development Program Report No. 100

August 2003

United States Department of the Interior
Bureau of Reclamation
Denver Office
Technical Services Center
Environmental Services Division
Water Treatment Engineering and Research Group

Mission Statements

U.S. Department of the Interior

The mission of the Department of the Interior is to protect and provide access to our Nation's natural and cultural heritage and honor our trust responsibilities to tribes.

Bureau of Reclamation

The mission of the Bureau of Reclamation is to manage, develop, and protect water and related resources in an environmentally and economically sound manner in the interest of the American public.

Disclaimer

Information contained in this report regarding commercial products or firms was supplied by those firms. It may not be used for advertising or promotional purposes and is not to be construed as an endorsement of any product or firm by the Bureau of Reclamation**.**

The information contained in this report was developed for the Bureau of Reclamation; no warranty as to the accuracy, usefulness, or completeness is expressed or implied.

Acknowledgments

Many different people contributed to the work in the report. At PerLorica's own office Aaron Jarrette was instrumental in writing the computer code necessary to implement the project's ideas, interface with the database, and make the outputs clear and readable. He also greatly contributed to this report. Nicholas Shahan was responsible for the web site development used for the outputs. Also thanks to William Conlon of Parsons, Brinkerhoff in Tampa Florida for his many helpful suggestions over the course of development.

Also, we would like to acknowledge and thank both Randale Jackson and John Walp at the Bureau of Reclamation (Reclamation) for their patience and counsel. Additional thanks to Michelle Chapman, Reclamation, for her careful and thorough proofreading and editing.

Finally, we particularly want to thank the Port Hueneme Water Agency (PHWA) for making all of the data from their 3 years of operation available for this work. Larry Mack, David Dean, and Gary Haggins at PHWA all lent their support. We also need to mention Joe Richardson, no longer with PHWA, but who gave generous and early support and encouragement in the development and installation of PL-Web software at the facility.

Table of Contents

List of Figures

List of Tables

List of Equations

Executive Summary

This project developed and refined methods for predicting the time to the next scheduled maintenance for membrane systems based on a number of different criteria. This "Days To Next Cleaning" (DTNC) as called herein is computed from historical data based on changes in key operating parameters applied to large numbers of normalized values. Data from a 3 year period, taken every 10 or so minutes, for a 1 MGD reverse osmosis system and a 1 MGD nanofiltration system, were used to validate and test the algorithms, which have proven to give useful predictive results.

Flow, pressure, conductivity, and temperature data from a membrane system stored in a database are used to normalize the instant performance back to a baseline condition. The normalization procedure uses standard ASTM methods; however, these methods are applied to each set of data points as they are stored in the database. Storing already normalized data gives the user a much more powerful tool to understanding what is happening in the membrane system at any given time. The program refined and developed for this work creates a trend line of the normalized data on daily or even more frequent basis. This trend "value" is calculated on varying look back parameters extending different time periods into the past. The calculated rate of change of each analyzed parameter – differential pressures, normalized permeate flow, and normalized salt passage – is then used to predict a DTNC.

The power of this approach is that the trend and predictions are made at least daily. This provides the user (and the software) with a 2^{nd} derivative type of prediction. In other words one can readily see, for example, a change in system differential pressure by plotting the rate of change of differential pressure against time, not simply the change over time in differential pressure itself as is commonly plotted. Of course a trained and experienced human observer can often spot developing problems from plots of raw data, however, the advantage of using a computer with a plethora of data points is the fact that relatively sophisticated analyses can be carried out quickly and efficiently on a regular basis.

The software was demonstrated on historical records from the Port Hueneme Water Agency's plant in Port Hueneme. Data at every 10 minutes starting in January of 2000 was available from a 1 MGD RO system operating at about 75% recovery and from a nearly identical system using NF membranes. Feed water came from shallow wells located in the nearby agricultural areas. The first 6 months of operation was plagued by bio fouling and this provided a fertile test bed for the algorithms developed in this work. The software predictions based on the algorithms developed produced reasonably close approximations to what actually happened.

Background and Introduction

It is well known that Reverse Osmosis (RO) and Nanofiltration (NF) systems can be very sensitive to changes in the operating environment. For example, temperature changes or feed water concentration changes can greatly affect either the operating pressure in constant flow environments or the flow rates in constant pressure environments. Most smaller membrane systems are operated in the constant pressure environment, while larger systems tend to be operated with flow set points. Rapidly and accurately normalizing the raw data back to initial conditions is important to diagnosing and dealing with problems before they become irreversible. Many of these systems run, for the most part, unattended and existing control systems, unless very, very sophisticated, are unsuited to performing this normalization task. This is normally left to the operators or to a subcontracted outside consultant. Many types of fouling can lead to a rapid decline in membrane performance and possible irreversible damage. The objective of this project is to refine existing methods for monitoring and fouling prediction, leading eventually to the deployment of a system that can accurately diagnose, and predict fouling, before it reaches the point of irreversibility.

RO and NF plant operators have long been accustomed to performing membrane maintenance operations in response to changes in the system's behavior. Cleaning of the membranes is indicating when fouling or scaling of the membrane surfaces results in rise of pressure differentials and / or trans-membrane pressures exceeding perhaps 10 to 25% of clean membrane conditions. Although the "trigger" or "set point" for such a condition can be as simple as a straight forward comparison of historical performance to current performance, the actual implementation of such an algorithm is complex.

PerLorica, since late 1999, has operated a commercial service for the operators of membrane based water treatment plants. This service, known as PL-Web™, or at present as WaterEye, (covered by US patents 6,332,110, 6,560,543, and others pending) collects data from the control system of the plant and forwards this data to a database residing at PerLorica's headquarters. The service provides alarm and process notifications to selected operators at the plant location or facility. One by-product of this service is that residing in the database are literally millions of data records chronicling the complete operating history of several plants extending back over three years. Port Hueneme Water Agency's (PHWA) plant in Port Hueneme was one of these plants. Data from a 1+ MGD RO system and a 1+ MGD NF system were available at approximately 10 minute intervals extending back to mid January of 2000. This data was used to develop, test, and validate the algorithms and code created for this project.

During much of the first half of year 2000, the plant experienced severe biofouling conditions and cleanings were carried out almost every several weeks. This provided a good test of the prediction capabilities in a period when the danger to the plant's membranes was high. In fact, many of the membranes needed replacing due to high differential pressure damage during these biofouling periods. Later in 2000 the plant changed from chlorination/dechlorination to chlorination with ammonia addition leading to chloramine formation. This change essentially

eliminated the biofouling and the plant has since run much more steadily. This provided an opportunity to test the methods against both rapidly fouling systems and against a stable operating environment at the same physical plant.

Conclusions and Recommendations

The usefulness of predictive software for monitoring RO plants has been extended and demonstrated in this work. By applying standard normalization procedures to large amounts of data, and carefully trending and extrapolating this data over varying time frames, useful predictive values can be obtained.

Using this software and getting the most benefits from application requires access to the data records of a membrane plant on a regular basis. Port Hueneme data is available to the program at 30 minute intervals, with three data sets sent each 30 minutes. This provides sufficient data to make accurate predictions and the frequency of analysis is sufficient to generate warnings or alerts well in advance of actual requirements for operator intervention. The data collection at Port Hueneme is fully automated and utilizes PerLorica's local data collection program and SQL database. This test can be continued indefinitely if desired.

Work Performed

The project's objective was to develop improved methods for handling membrane plant data and to develop sufficiently robust algorithms to enable detection of incipient scaling or fouling well in advance of immediate needs. This was accomplished through applying a variety of analytical approaches to normalized membrane plant operating data and extrapolating the rate of change to predict a "Days to Next Cleaning" (DTNC) value based on normalized permeate flows, differential pressures, and normalized salt passage.

To get to the point that a prediction can be made, however, requires significant data manipulation, particularly when large numbers of data points are used. Consider that the look back periods involved for calculating a Days to Next Cleaning (DTNC) value are 7, 20, and 30 days respectively. 144 instances of normalized flow are recorded daily per membrane system train. So to look back the three periods requires performing a linear regression analysis, a standard deviation calculation, a "goodness of fit calculation" (R^2), a slope and an intercept calculation for 2016, 5760, 8640 data points consisting a date/time and value. This calculation is redone once the first standard deviation is calculated to bypass "bad" or out of range data. The power of the system is that this is provides a much more powerful tool for seeing changes in membrane systems as the change propagates through the system.

Raw Data Collection

Port Hueneme Water Agency graciously consented to allow use of their historical data for this project. Table 1 illustrates the basic raw data collected for the RO and NF systems. Additional data is also collected for PHWA that is useful to PHWA but not relevant to this project. This includes data about the EDR system in operation at PHWA.

Data is collected at PHWA directly from the system's PLC's via RSLINX™ (Rockwell Software). RSLINX software was provided by Port Hueneme and is also used by the PHWA control system to access the local PLC's (PLC 5's by Allen Bradley). PerLorica originally wrote and provided to PHWA the local data collection software, written in Visual Basic 6.0. This software, designated PL-Access, originally used ftp protocols to send data to an ftp server located at PerLorica's HQ. Later versions were written in Microsoft Visual C++ and use a direct ADO (Active Data Objects) connection to a SQL 2000 server, also located in Grass Valley. The PHWA computer being used for this application runs Windows NT and would require upgrading to utilize the latest version of PL-Access. Therefore the current program at PHWA uses FTP to send data to the FTP server. At the FTP server the raw data is collected and put into the SQL database by the FTP server computer running Windows 2000 server. Figure 1 shows how the typical PL-Access system can be configured. At PHWA the PL-Access system accesses point and register data from the PLC's only although it can access any source of data from instruments to SCADA systems to Historians to other computers.

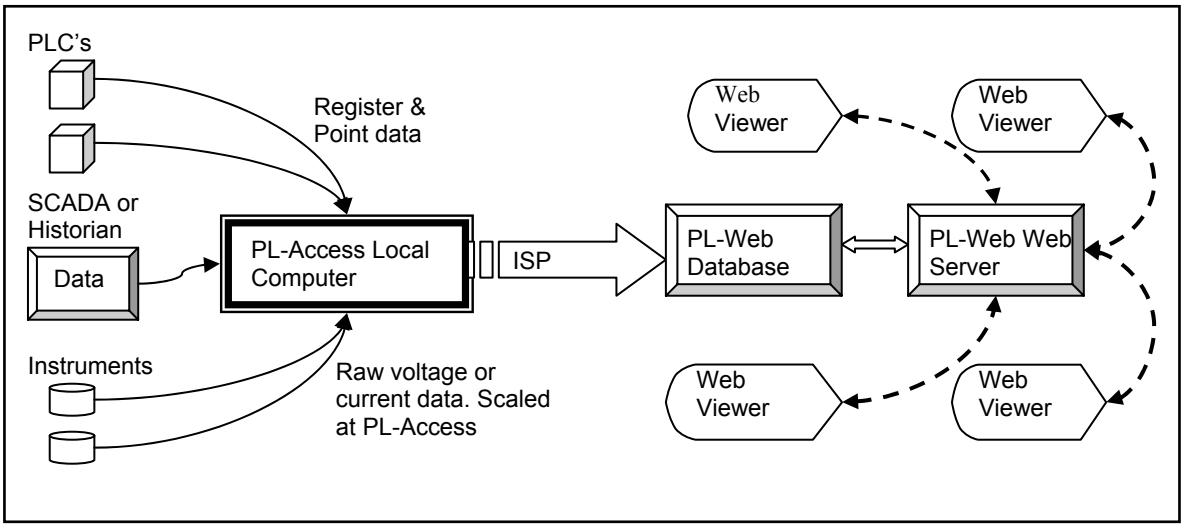


Figure 1.—PL-Web Functions.

Access to PLC data and SCADA data is usually through the DDE interface, although OPC can be used as well. In the case of PHWA, RSLINX acts as the DDE server to PL-Access as well as to the SCADA system in use at PHWA. PL-Access is configured to wait 1.0 seconds between data requests so that little additional data load is imposed on the system. Typically a SCADA or HMI system might query all available data (hundreds or thousands of data items) every second, so the load imposed by PL-Access is quite minimal.

At the PHWA site, PL-Access accesses some 74 data items at a typical capture frequency of every 10 minutes. This varies somewhat since the timer is suspended while the actual data transfer is in progress, so the average frequency is typically 10 minutes plus 15-25 seconds. Data is sent to the SQL server via ISP dial up every 30 minutes.

Once the raw data is captured, the calculated or derived data is immediately generated by the Server running the database. These data are referred to as virtual sensors and are displayed in light cyan color on the web site. Predictive data, that is data calculated from a full prior data set, is calculated daily at about 12:00 AM. The derived and predictive data calculated by the database for this project are listed in Table 2.

The raw data collected from PHWA is basically available at 10 minute intervals starting at January 11, 2000 through the present. In the course of this project, the derived data has been recalculated multiple times and the current versions are now present over the same time period. Since the data is readily available over the web, this allows comparisons of the predicted cleaning times with the actual cleanings carried out by PHWA over the time period. The author was physically present at several of these cleaning procedures in the mid 2000 time period and assisted PHWA with the cleanings.

Normalization Methods

The first versions of PL-Web™ used normalization methods derived from ASTM D4516.[1] However, these were much simpler than the full normalization and in addition were the inverse of the normal ASTM method. Our original approach was to compare "what the system should be producing" at the given conditions to what the system is actually producing. The method used essentially the same equations as the ASTM method, but slightly re-arranged. This gives a normalized performance line that may or may not be a straight line, but when plotted against the actual conditions shows up deviations quite readily. However, at times this caused confusion interpreting the data. Therefore, even before this project was awarded, we switched to the more conventional normalization approach specified by ASTM 4516.

ASTM D 4516-00 Method

The normal ASTM method calculates "normalized" permeate flow rates back to some standard set of conditions. This is normally based on 25 deg C and the start up conditions of the plant relative to applied pressure and osmotic pressure conditions. The basic formulas for flow normalization are simplified below.

Equation 1 – Normalized Flow, ASTM 4516-00

$$Q_{ps} = Q_{pa} * \frac{\left[P_{fs} - \frac{\Delta P_{fbs}}{2} - P_{ps} - \pi_{fbs} + \pi_{ps} \right] (TCF_s)}{\left[P_{fa} - \frac{\Delta P_{fba}}{2} - P_{pa} - \pi_{fba} + \pi_{pa} \right] (TCF_a)}$$

Where

- Q_{ps} is the product normalized flow
- Q_{pa} is the actual measured product flow
- P_{fa} is the applied actual pressure and P_{fs} the applied standard/baseline pressure
- P_{pa} is the applied actual permeate pressure and P_{ps} the applied standard/baseline permeate pressure
- ΔP_{fba} is the actual differential pressure and ΔP_{fbs} the standard/baseline differential pressure
- π_{fba} is the osmotic pressure of the actual feed/brine average and π_{fbs} the osmotic pressure of the standard/baseline feed/brine average
- π_{pa} is the osmotic pressures of the actual permeate and π_{ps} the osmotic pressure of the standard/baseline permeate
- TCF_a is the temperature correction factor at the actual temperature and TCF_s is the factor at the standard temperature. N.B. is 25 C and therefore TCF_s is usually = 1.0.

Equation 1 can be simplified in appearance:

Equation 2 – Simplified Normalized Flow, ASTM 4516-00

$$Flow_{Normalized} = Flow_{Actual} * Temp\,Correction * \frac{TransMembrane\,PSI_{Baseline}}{TransMembrane\,PSI_{Actual}}$$

Note that with this method, constant actual permeate flow with INCREASING actual pressure results in a DECREASING normalized permeate flow. The ASTM methodology normalizes the permeate flow for pressure and temperature. This means that if a higher pressure is required to produce the same permeate flow rate, then the normalized permeate flow has decreased. For our analyses, this simply means that when normalized flow decreases below the set percentage of baseline normalized flow, the time for cleaning is at hand. Typical RO operating guidelines call for cleaning when normalized flow decreases by about 10-15% from clean values.[2]

Temperature correction methods vary by membrane manufacturer, but always result in decreasing corrected flow with increasing temperature. For PHWA we used an industry standard approach:

Equation 3 – Temperature Correction Factor

$$TCF = \exp\left(MembraneTempCoeff * \left(\frac{1}{298.16_{(25DegC)}} - \frac{1}{DegK_{actual}} \right) \right)$$

The Membrane Temperature Coefficient may be stored in the database for each system. The typical value for the membranes at PHWA is around 2700 to 3100. Thus at 25 deg C, TCF = 1.0. At 30 deg C, TCF is 1.16 to 1.18. The simplified ASTM method ($1.03^{(T-25)}$) gives 1.16 at 30 deg C. The coefficient used for the PHWA data is currently set at 3020.[3]

Trans-Membrane Pressure calculations need to reference the osmotic pressure of feed, concentrate, and permeate as well as the differential pressures. Typically only conductivity data is available from a system, so some equation for converting conductivity to osmotic pressure is needed. The full ASTM 4516 method dictates an ion by ion calculation to get osmotic pressure. This is typically never available on a regular basis, although a full analysis of the feed is often available every year or so. Fortunately, conductivity for feed, permeate and concentrate is usually available and, when combined with temperature, a reasonable guess of osmotic pressure can be made. If the guess methodology is consistent, then small deviations from actual osmotic pressure make little difference to the normalization calculations.

ASTM specifies an equation for osmotic pressure based on equivalent NaCl concentration in the average feed water. To convert electrical conductivity in micro Siemens (uS) per cm to equivalent ppm of NaCl we have used several empirical equations extracted from the various literature and manufacturer sources.[4,5,6] In general PHWA feed water can be characterized as brackish water. Feed water conductivity is in the range of 1250 to 1400 μS/cm). Permeate conductivity is reduced to about 30 to 60 μS/cm for the RO system and about 100-200 μS/cm for the NF system. We used slightly different equations for permeate versus feed and concentrate conversions to TDS because the permeate has somewhat higher NaCl / TDS ratio than the feed

or concentrate. The general form of the equation assumes that the conductivity instrument has already corrected the conductivity back to 25 deg C.

Equation 4 – Conductivity to TDS NaCl

$$TDS_{as\ NaCl\ \ ppm} = A*(\mu S/cm)^{B}$$

For feed and concentrate we used A = 0.5604 and B = 1.0296[7] while for permeate the values are A = 0.4296 and B = 1.0188 respectively taken from a curve fit to standard data. These give good correlation to commercial RO normalization programs such as Hydranautics™ RODATA program. Figure 2 illustrates.

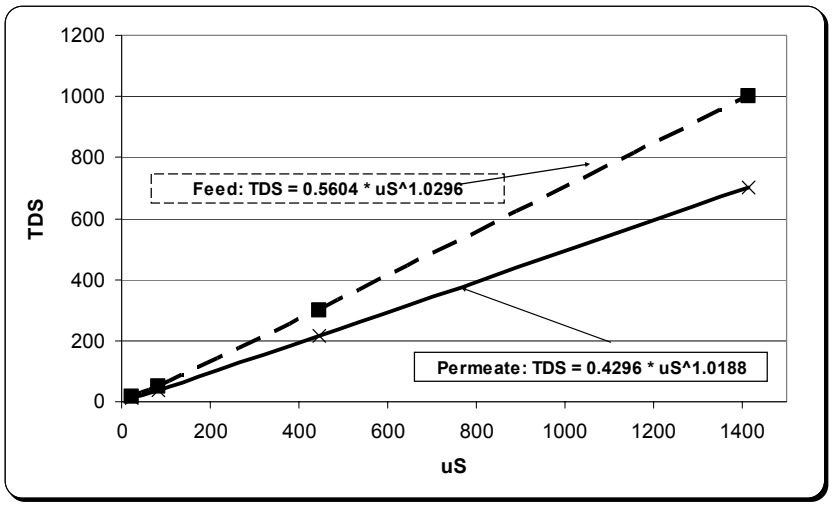


Figure 2.—TDS vs. Conductivity.

To get from a TDS as NaCl value to an osmotic pressure, we utilized the existing ASTM formula for the feed water. While this introduces some error compared to more rigorous calculations made by ion, most systems do not operate with widely varying ionic compositions. Since we are more interested in changes in performance rather than absolute performance numbers, small errors have little affect on the accuracy of the predictions. For permeate osmotic pressure ASTM specifies a fraction of the feed osmotic pressure, while we used the more accurate curve fit data for TDS from conductivity applied to the basic equation (equation 5) for osmotic pressure, giving slightly more rigorous results than ASTM in some cases.

The ASTM 4516-00 equation (Eq 8) for osmotic pressure based on feed brine average concentration in psi is:

Equation 5 – Osmotic Pressure

$$\pi = 0.2654 * C_{fb} * \left(\frac{DegK}{\left(1000 - C_{fb} / 1000\right)} \right) * 0.145 \left(psi / kpa \right)$$

As is evident, performing these calculations for each data point taken at 10 minute intervals requires some substantial computing power as well as access to stored constant values and multiple data points. For this reason, the control systems for most small and medium size membrane plants do not calculate real time values of normalized performance. Instead, typically a managerial operator downloads raw data into a spreadsheet and makes a manual / visual analysis of current performance.

Salt Passage Normalization

Equation 6 – Salt Passage Normalization ASTM 4516-00

$$\%SP_s = \frac{EPF_a * STCF_a * C_{fbs} * C_{fa}}{EPF_s * STCFs * C_{fba} * C_{fs}} * \%SP_a$$

Where

- C_{fbs} is the standard/baseline log mean concentration of the feed and brine as ppm NaCl
- C_{fba} is the actual log mean concentration of the feed and brine as ppm NaCl
- C_{fs} is the standard/baseline concentration of the feed as ppm NaCl
- C_{fa} is the standard/baseline concentration of the feed as ppm NaCl
- EPF_a is actual average permeate flow
- EPF_s is average permeate flow at standard/baseline conditions
- al permeate pressure and P_{ps} the applied standard/baseline permeate pressure
- $STCF_a$ is the salt transport temperature correction factor at the actual temperature and $STCF_s$ is the salt transport factor at the standard temperature. N.B. is 25 C and therefore $STCF_s$ is usually = 1.0.

The program also does salt passage normalization according to ASTM 4516-00 as well. With salt passage it is easier to appreciate changes than with rejection since the apparent percentage changes are so much greater. The method is quite similar to flow normalization, except that a baseline feed conductivity is required and the temperature correction factor may use different constants from the flow temperature correction factor.

Differential Pressure

Although there is no specification for differential pressure (DP) normalization in ASTM 4516, some membrane manufacturers implement a version based on element specific conditions, usually in the form of ΔP = C * Flow ^ B, where B is typically 1.45 to 1.6[8], and C is a coefficient specific to each element. This reduces to:

Equation 7 – Normalized Differential Pressure Loss

$$\Delta p_{Normalized} = \Delta P_{actual} * \frac{F_{fbs}^{1.5}}{F_{fba}^{1.5}}$$

Where

- F_{fbs} is the average feed-brine flow at standard/baseline conditions
- F_{fba} is the actual average feed-brine flow

The coefficient, "C" varies widely from element type to element type. In addition, systems rarely have flow data for the two stages measured separately, thus this equation becomes of minimal use. Since most membrane systems operate at constant flows and recovery, we have for the most part eliminated any ΔP normalization considerations and instead measure absolute values. Differential pressure changes in themselves are indicative of fouling and scaling so we have accommodated systems with two "stages" of pressure measurements. At PHWA the system incorporates pressure instrumentation for both stages of both systems. With ΔP from both stages it is possible to differentiate between scaling and fouling. A rising ΔP in stage 1 is indicative of fouling by suspended solids or biological growth. A rising 2nd stage ΔP is indicative of scaling, due perhaps to changing conditions, recovery, or loss of scale inhibitor. This also suggests that if the actual scale inhibitor dose of active ingredient could be measured and monitored the concentration could be correlated to changes in 2nd stage DP, resulting in the ability to optimize the dose just necessary to prevent scaling. Note the monitoring must be of the "active" ingredient or other species that exactly tracks the active ingredient, not simply the mass flow rate.

Constant Values

An inspection of Equation 1 shows that a number of constants, unique to the system being analyzed, require storage or recall for each data point calculated. At present we store the following data items for each system:

- Baseline Trans-Membrane Pressure (Top term in Equation 5)
- Baseline Differential Pressure (ΔP_{fbs})
- Baseline Feed Conductivity (C_{fs})
- Baseline Feed Flow
- Baseline Recovery
- Temperature correction coefficient for flow
- Temperature correction coefficient for salt passage

Unfortunately many systems have no readily available data to support the required constants. Therefore, part of the work in this project was to develop a technique to "reset" the baseline values based on performance over past periods. The idea is that the software can "reset" itself to a new baseline if conditions change dramatically. This is discussed in detail below.

Calculation Scale Up

Application to tens or in the case of PHWA millions of data points requires that the system store most of the calculated values. It is unrealistic to re-calculate all of the normalized data each time

a system analysis is performed. We investigated several possibilities in this respect, including storing various intermediate results. We performed benchmark studies using a Dell PowerEdge Server running Windows 2000 Server, SQL Server 2000, 750 MB of memory, and a single 1.0 GHz processor. To recalculate all of the normalized data, for all of PHWA's approximately 370,000 data sets, required 2 to 3 hours. However, when the statistical calculations described below were added, the time required rose to an overnight run. Based on these tests, we concluded that executing the normalized calculations on every data set as the data arrived was feasible, provided the statistical calculations were only done once a day at midnight or 1:00 AM, times when the server use was minimal. Thus the current system processes data looking back 30, 20 and 7 days for both differential pressures, the normalized permeate flow, and the normalized salt passage. These calculations require only a few minutes of server time.

Description and Methodology

Since the data from PHWA was flowing into a production server, PL-Web now called WaterEye™, we needed to carefully test any changes and make any changes incrementally without affecting the production server.

The basic program flow we have had in place for PL-Web for some time is illustrated as follows:

- Data arrives at server from PHWA
- The data is stored in appropriate table
- A Stored Procedure in SQL calls programs written in Visual Basic or C++ or other language. These programs may be resident on this server or elsewhere on the network or even remote.
- These programs:
    - Run data analysis calculations such as comparing actual values to High-High and/or Low-Low set points.
    - Calculate derived data such as normalized flow, differential pressure, recovery, salt rejection, and so on.
    - Store the derived data in the appropriate table in the database
    - Issue alerts/warnings if necessary.
    - Write notes to the system log

At midnight each night, the server calls an additional stored procedure. This stored procedure in turns calls the programs refined under this contract - namely, the programs that perform a "look back" function to calculate trends and conditions in the membrane plant. The results are also stored in database, enabling the ability to "trend the trend" if necessary since this is computed *de novo* each day.

A web user can then query the database from the PL-Web web site to create graphs or reports as needed. The web page requests various stored procedures from the SQL database which create reports (Excel, HTML, and PDF), graphs, or tables as necessary which are displayed for the user. Appendix 1 contains an extensive set of screen captures showing data displays available to the user.

The minimal data set required is shown in figure 3:

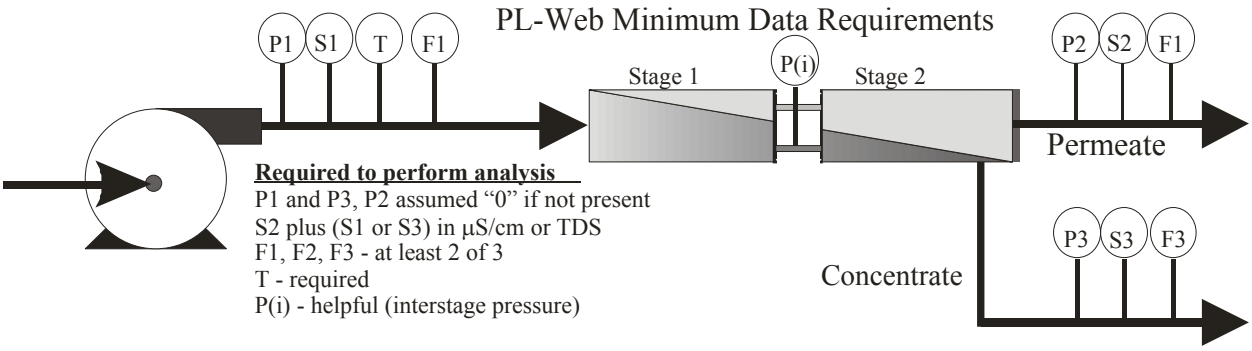


Figure 3.—Minimal Data Set.

Methodology

We tried a variety of computational approaches and varying "look back" periods. It quickly became obvious that we would need to actually evaluate varying time periods each time we performed the analysis. Consider the following illustration of ΔP v Time. (See figure 4.)

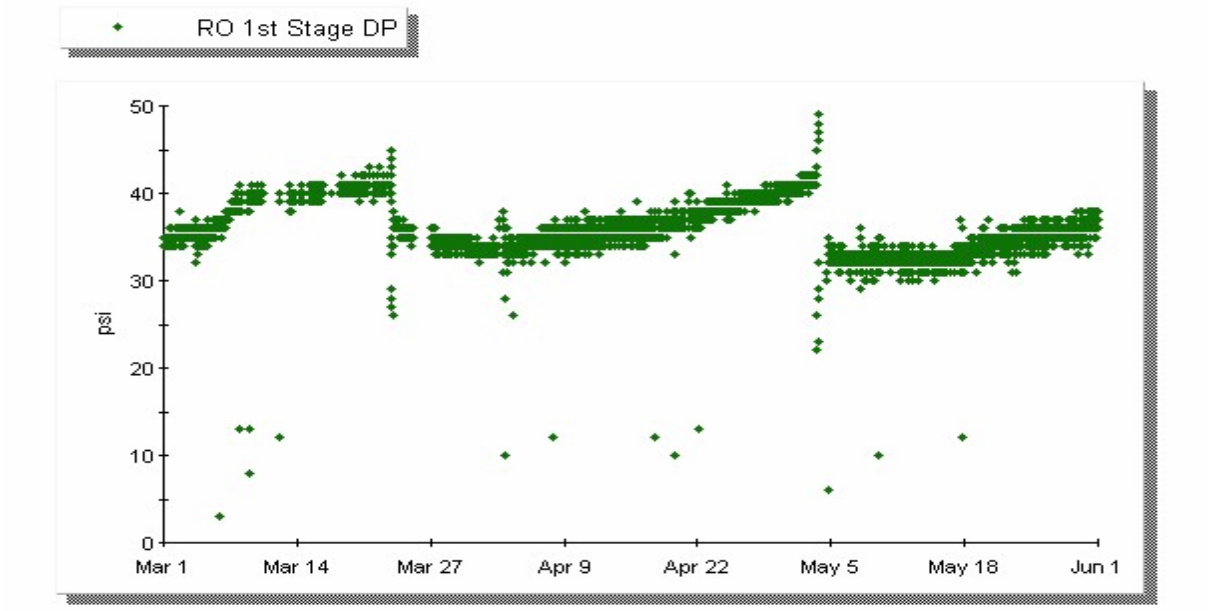


Figure 4.—ΔP vs. Time, 3/1/2000 to 6/1/2000 .

Note that this data, taken from actual PHWA RO performance has three distinct steps. These "steps" occurred due to cleaning of the system, with clear recovery of some of the lost DP. A plot of normalized flow over the same period gives a similar result. Plotting a trend over the entire time period provides a completely erroneous result. Consider the plot of the same data with trend lines added as shown in Figure 5.

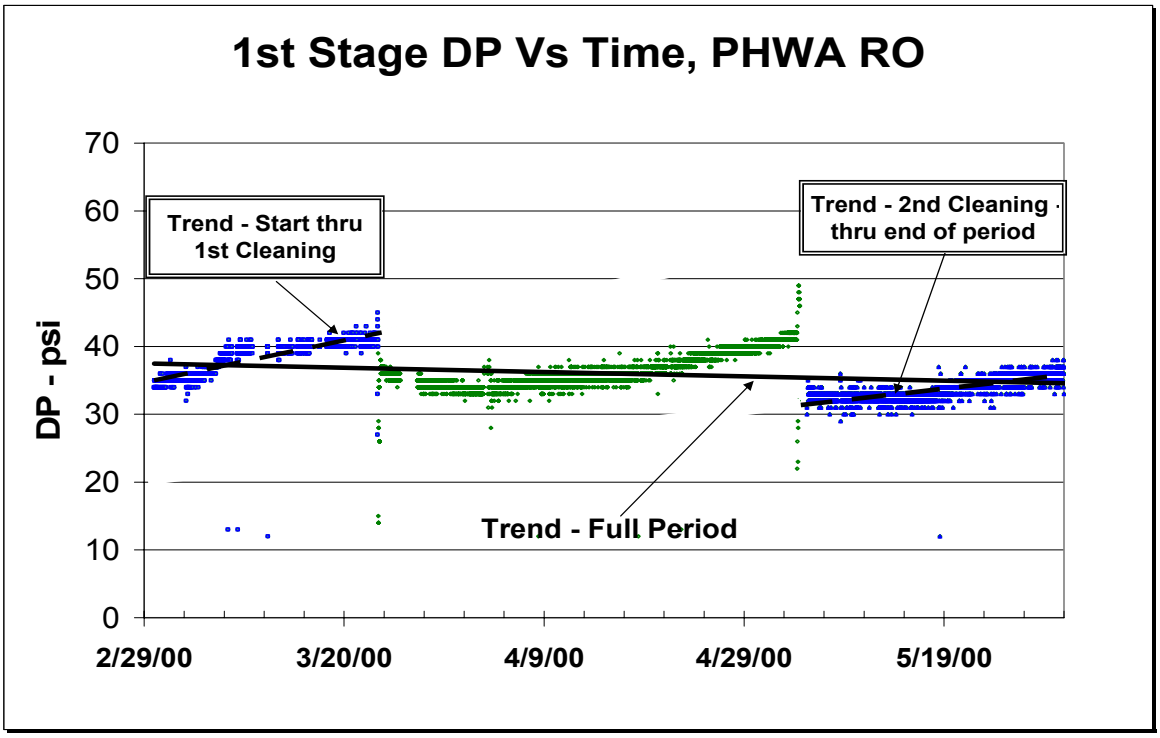


Figure 5.—Trend Lines.

The full period trend line indicates a decreasing ΔP even though at the end of the period the ΔP is clearly increasing rapidly.

This chart already has excluded data that is zero or near zero. The scattered low value data points come from ΔP values recorded during cleaning or during actual startup and shutdown periods. ΔP when the system is offline is still recorded and typical values are 1 to 2 psi – probably because of water left in the lines to the gages or slight differences in calibration between the gages used for the calculation of ΔP. If this data is included in the trend lines then additional error is introduced.

As a result, in order to make the most accurate prediction possible we analyze the data over three different time periods: 1) Data for the past week. 2) Data for the past 20 days. 3) Data for the past 30 days. This allows us to monitor how the system has been performing over the past month as well to focus on the last week of operation. If conditions in the system have been gradually changing, then analyzing data for the past month will usually offer the best results. However, if the system's performance is changing rapidly, then focusing on the past week often gives best result.

Figure 6, taken from an actual web page plotted for this report, illustrates the computed "days to next cleaning" (DTNC) based on the first stage ΔP. The DTNC values are roughly airbrushed in as a line since the point values are not distinguishable in black and white. Note that the DTNC values hit zero about a week or so before the system was actually cleaned and then rise to more

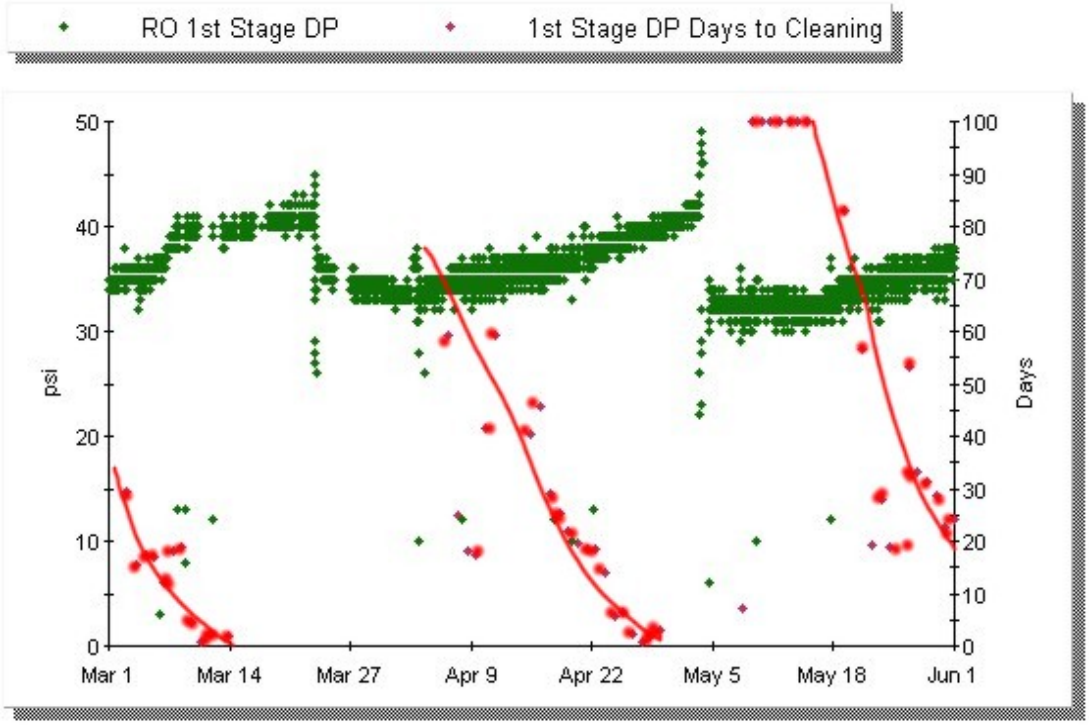


Figure 6.—Web Plot, DP Days to Next Cleaning.

than 100 days immediately thereafter. Figure 7 shows some of the same data over the full month of March. Figure 7 plots data every 4 hours so some of the peaks and valleys of Figure 6 may be missing. The plant was clearly experiencing an increasing 1st stage ΔP and the DTNC falls rapidly. The system will alert when the DTNC value falls below a preset point, usually 7-10 days before cleaning is needed, however this can be reset easily by the plant owner via a web page.

Once "cleaning occurred" and the system returned to normal operations for a few days, the DTNC value rose back to the default maximum of 365 days. In this particular case, bio fouling always took a few days to build back up and become noticeable again, so during these first few days the ΔP did not rise.

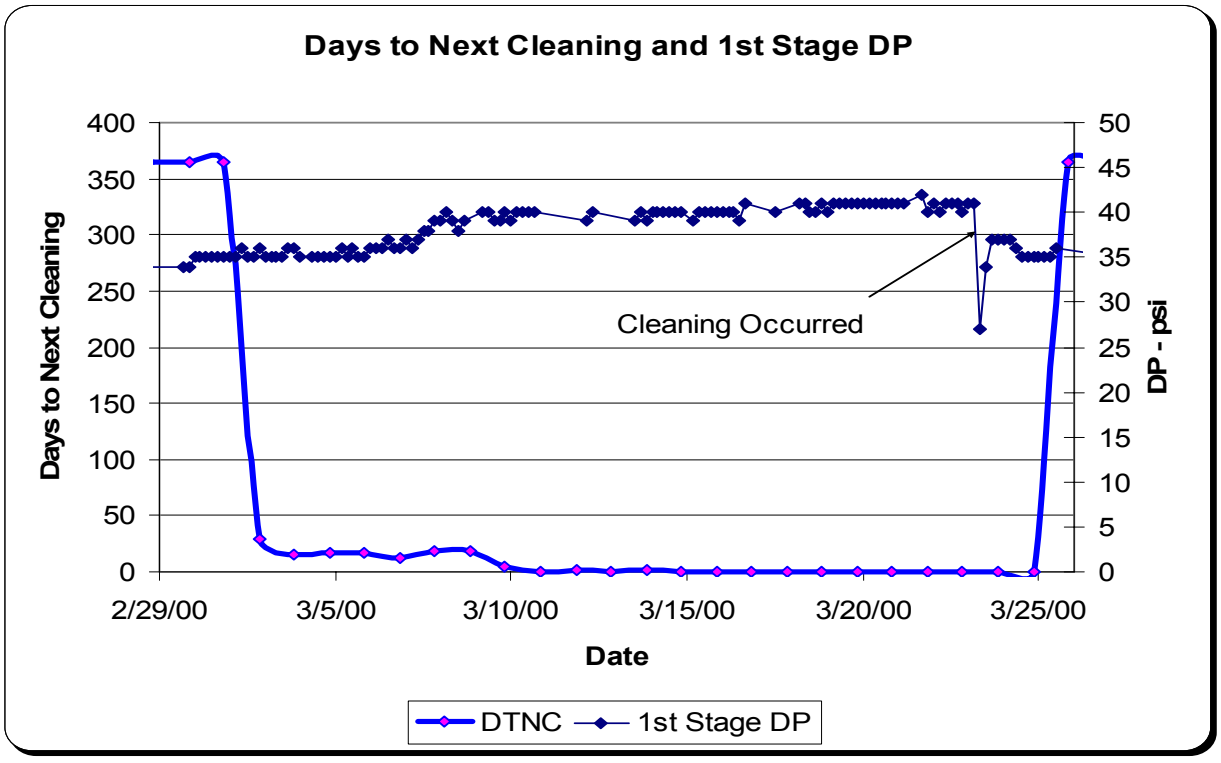


Figure 7.—DTNC by 1st Stage ΔP Detail.

Figure 8 plots the same time frame as Figure 7 but shows the normalized flow and Days to Next Cleaning as well based on Normalized Flow. Figure 9 is the same but for 2nd Stage ΔP.

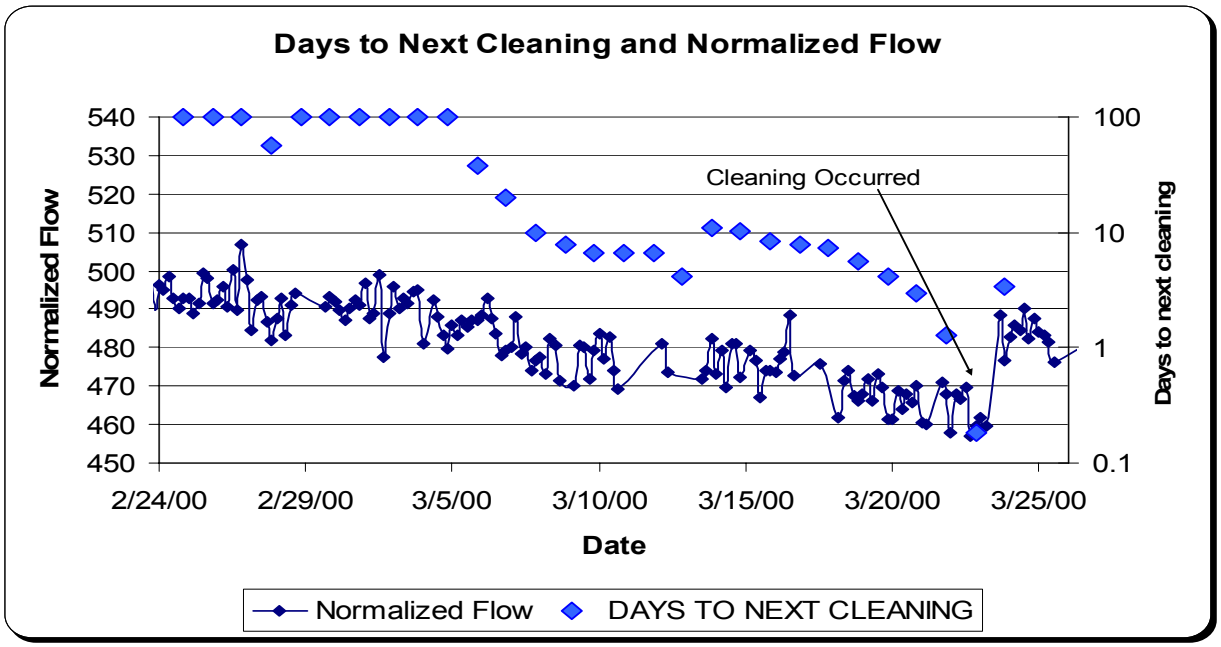


Figure 8.—DTNC from Normalized Flow.

Note that the 2nd stage ΔP (Figure 9) was almost completely straight lined, indicating no scaling or fouling in these latter elements. The DTNC stayed at 100 days or above. 100 days is the plotted value if the calculation reports more than 100 days.

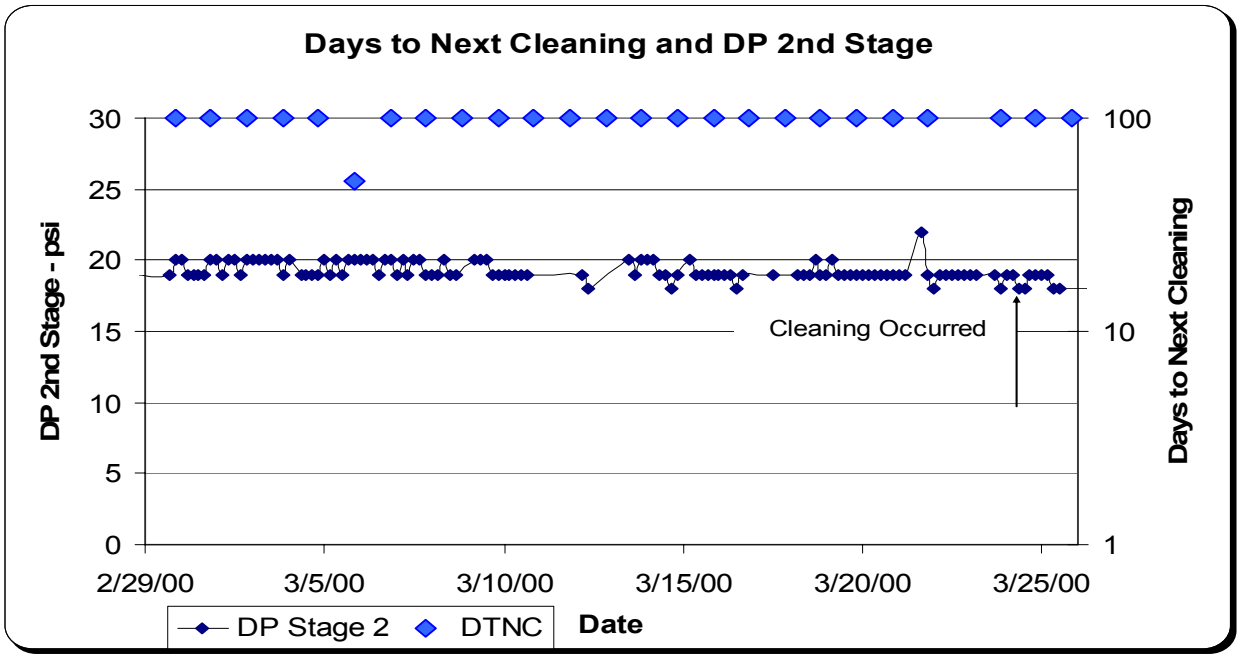


Figure 9.—2nd Stage DP.

Analytical Techniques – Days to Next Cleaning

Our system currently performs statistical analysis on five measured variables of an RO or membrane system: They are:

- 1st Stage ΔP
- 2nd Stage ΔP
- Overall System ΔP
- Normalized Permeate Flow
- Normalized Salt Passage.

The internal Program Flow is ordered as follows:

1. Get Configuration Data
2. Get the Complete Set of Data from the Database Over the Period
3. Perform Linear Regression on 1st Set Data
4. Check the Data's Goodness of Fit
5. Get a 2nd Set of Data if Necessary, Consisting Only of Data within One Standard Deviation of Regression Line

6. Perform Linear Regression on 2nd Set Data
7. Predict Days to Next Cleaning (DTNC) Based on Linear Regression Analysis
8. Check the Confidence of our Results
9. Update Database

Get Configuration Data

During this step the program gets the sensor data that will be used to calculate the DTNC. Currently, we can make cleaning predictions based on five factors:

1. 1st Stage ΔP
2. 2nd Stage ΔP
3. Overall System ΔP
4. Normalized Permeate Flow
5. Normalized Salt Passage.

This sub module computes a value of DTNC using each of the five data items above. This value is based on taking the baseline data for each sensor and multiplying it by some setpoint or factor that is configurable (i.e. if the system should be cleaned when the ΔP increases 15% over the baseline, then the factor would be 1.15).

It computes the DTNC for each of three "look-back" periods which form the time base of our linear regression calculations. Currently, the three look back periods are 30, 20 and 7 days. The look back period that produces the shortest number of days to the next cleaning, at an acceptable confidence level, is used. Thus some three times five, or fifteen DTNC's are computed.

We also compute the confidence level required to make an accurate prediction (currently 80%). If we are unable to make a prediction at this confidence level with any of the three look back periods, the confidence level is cut in half and we try to make predictions again. If the system is unable to make a prediction at this confidence level, then the system fails to make a prediction. A failed prediction is most likely the result of very scattered or inconsistent data in the database in which no recognizable pattern could be observed.

Details of each step follow below. These are reflected in the Visual Basic source code included in the appendices. Statistical equations were taken from standard Excel[9] functions.

Get 1st Set (Pass through the data collection – not membrane Pass) Data

This gets the data that has been collected for a given sensor during the three look back periods ignoring data that is less than or equal to zero.

Perform Linear Regression on 1st Pass Data

Using the data from the 1st pass data collection, this section of the program performs four linear regression functions on the data:

GetSlope

This module calculates the slope of the line based on the known X's and Y's. The slope is defined as the rate of change along the regression line. The slope is calculated based on the following equation:

Equation 8 – Regression Line Slope

$$b = \frac{n\sum xy - \left(\sum x\right)\left(\sum y\right)}{n\sum x^2 - \left(\sum x\right)^2}$$

GetStdDev

Calculates the standard deviation.. The standard deviation is based on the following equation:

Equation 9 – Standard Deviation

$$\sigma = \sqrt{\frac{n\sum x^2 - \left(\sum x\right)^2}{n(n-1)}}$$

GetYIntercept

Calculates the Y intercept. This is the point at which a line will intersect the y-axis based on a best-fit regression line. The Y intercept is calculated based on the following equation:

Equation 10 - Slope

$$a = \overline{Y} - b\overline{X}$$

GetRSQ

Calculates the value R^2. This is used to determine how confident we are in the slope we calculated above. R^2 is calculated based on the following equation:

Equation 11 – R^2, The Goodness of Fit

$$R^2 = \left\langle \frac{n\left(\sum XY\right) - \left(\sum X\right)\left(\sum Y\right)}{\left(\left[n\sum X^2 - \left(\sum X\right)^2\right] * \left[n\sum Y^2 - \left(\sum Y\right)^2\right]\right)^{0.5}} \right\rangle^2$$

Check the Data's Goodness of Fit (R^2)

This sub checks to see if the data collected in the 1st pass is sufficiently uniform to make an accurate prediction. If one of the following two conditions is met, then we have sufficient data:

1. If the standard deviation divided by the average of the Y's is less than or equal to 3%. This indicates that we have a flat line.
2. If R^2 is greater than or equal to 90%. This indicates that we have a line without much scattered data.

Get 2nd Set (Pass) Data (if necessary)

If it was determined that the data collected in the 1st Set (Pass) was not sufficient to make a prediction of DTNC, then we get data for all three look back periods that is within one standard deviation of the average data for each period. For example, if the average of the data was 30 and the standard deviation was 5, then we would get all data that was between 25 and 35.

Perform Linear Regression on 2nd Pass Data

Perform the same linear regression analysis that was done on the 1st pass data using the 2nd pass data.

Predict Days to Next Cleaning Based on Linear Regression Analysis

For each of the look back periods, we predict when the system needs to be cleaned. By using the slope of the line and the Y intercept, we can predict when the value of the current sensor will hit the level at which the system needs to be cleaned. The number of days to the next cleaning (based on the current sensor) is calculated using the following formula:

Equation 12 – Days to Next Cleaning

$$Days\,To\,Next\,Cleaning = \frac{Clean\,Value - Y\,Intercept}{Slope} - Look\,Back\,Period$$

Where CleanValue is the baseline data for this sensor multiplied by some factor (i.e. if the system should be cleaned when the ΔP increases 15% over the baseline, then the factor would be 1.15), Y Intercept is the value at which the line crosses the y-axis, Slope is the slope of the line and Look Back Period is the number of days for the current look back period.

Check the Confidence of our Results

For each of the results returned by the three look back periods, we check to see how confident the program is in its predictions. For the results to be accepted, the standard deviation must be greater than zero and one of the following conditions must exist:

1. R^2 must be greater than 80% and the standard deviation divided by the cleaning level must be less than 15%. If R^2 is high, then the program is confident in its results. However, if the standard deviation is exceptionally high (>15% of the cleaning level), the resulting R^2 will be high even though the data is quite scattered.

2. The standard deviation divided by the average of the Y's must be less than or equal to 3%. If the standard deviation is very small in relation to the average of the Y's, this indicates that the data fits nicely on a line (i.e. it's not scattered) and the program is confident in its prediction. However, when the standard deviation is very small in relation to the average of the Y's, R^2 may also be less than 80%, which results in condition (1) failing the test. Fortunately this combination does not occur frequently in typical operating conditions.

After cleaning predictions are made for all three look back periods, the one that predicts the fewest number of days will be accepted. However, if none of the three look back periods can make a prediction with acceptable confidence, the requirements for acceptance are lowered (R^2 must then be greater than 40%) and all three look back periods are checked again. If the program fails to make an acceptable prediction at this confidence level, no prediction is accepted because of insufficient data. The program reports "not available" to a web inquiry.

Update Database

If the program was able to make a prediction with an acceptable confidence level then the number of days that was predicted gets stored in a database. The database keeps track of the number of days each sensor predicted and the stores the lowest (soonest) DTNC reported from the sensors. This allows one to see how a specific sensor (or calculated value) is being affected by the conditions of the membrane or how the system as a whole is being affected by the conditions of the membrane.

Base Line Auto Reset

Occasionally conditions in an RO or membrane system change resulting in new baseline data. If the prediction software continues to use baseline data that is no longer accurate, it will be unable to make accurate predictions on when to clean the membranes. Therefore, it is important to occasionally check to see if the baseline data on record is still accurate, and update it if conditions have changed. We eventually settled on executing the "check" once a week.

Program Flow

- Get Data
- Calculate New Baseline
- Update Baseline if Necessary

Get Data

This module collects the current baseline data for each of the sensors used in the prediction software. Then it collects the values of the sensors during the past week. To get the most accurate results, data that is collected while the RO or membrane system is off, is ignored. Also, any data that is less than 50% of the old baseline is ignored.

Calculate New Baseline

The new baseline is calculated by taking the average of the data collected in the past week. If the standard deviation from the average is very small (<2.33% or ½ sigma) then we know that we have a flat line and we can potentially use this value as the baseline.

Update Baseline if Necessary

If the newly calculated baseline is within 10% of the old baseline, we will continue to use the old baseline. If the difference between the old and new baseline is greater than 10–15%, or the value specified in the database, then the baseline data is updated.

Results

We can conclude that with sufficient operating data from a membrane treatment plant, automated analysis software can do quite a reasonable job of alerting operators to impending needs for service.

PHWA provided an excellent example, as it was a system that experienced both upset conditions and long periods of stable operations.

Compare the two periods of Feb-May 2002 to Feb-May in 2000 using actual web graphs. In the later periods, the normalized flow is nearly constant. The days to next cleaning never goes below 30 days, which is effectively indicating a very constant performance. (See figure 10.)

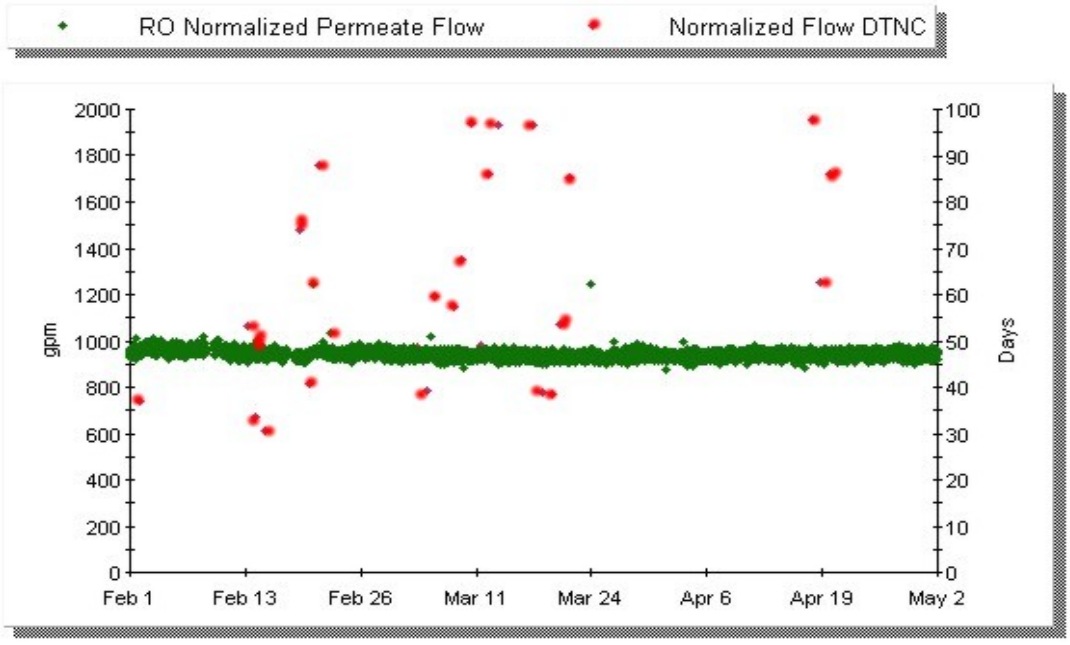


Figure 10.—Feb-May 2002, Norm Flow, DTNC.

The same time frame, plotted in figure 11 for 1st stage ΔP and DTNC shows the same flat lines and high DTNC values.

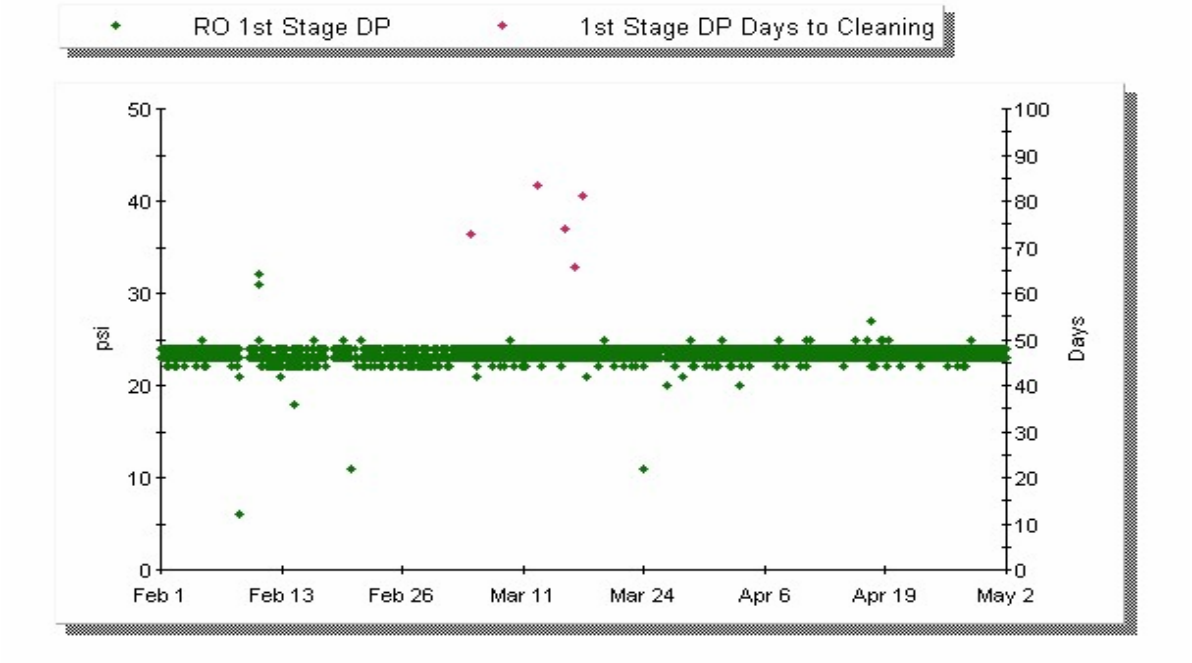


Figure 11.—Feb-May 2002, ΔP and DTNC.

Compare this to the Feb-May period in 2000. The DTNC is airbrushed in to make it easier to see. DTNC values of greater than or equal to 100 days do not show on the chart. (See figure 12.)

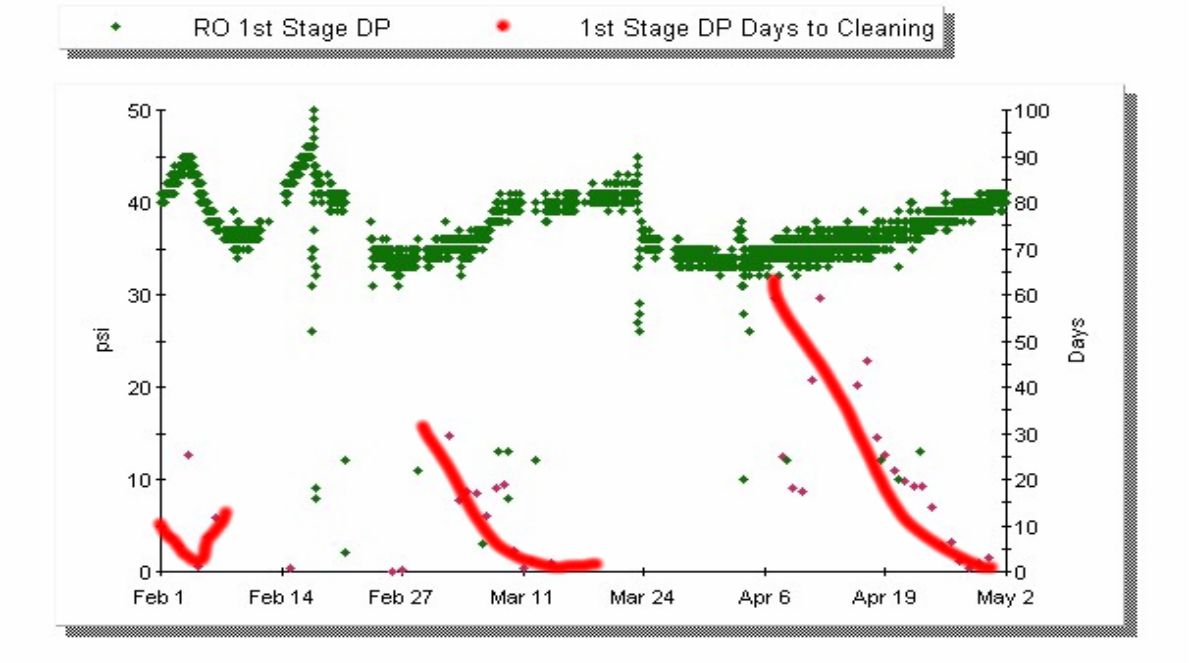


Figure 12.—Feb-May 2000, DP and DTNC.

Around June – July of 2000 PHWA replaced some NF membranes and introduced chloramination as the primary disinfection process. Prior to this time, the disinfection was accomplished solely through chlorine injection then subsequent dechlorination with bisulfite. Chloramination removed the requirement for dechlorination and it was now possible to carry a small positive residual of chloramines throughout the system. This quickly brought the bio-fouling issues under control. As a point of interest, the change was so dramatic that the cost to run the plant dropped substantially. The last two figures in this section illustrate. (See figures 13 and 14.)

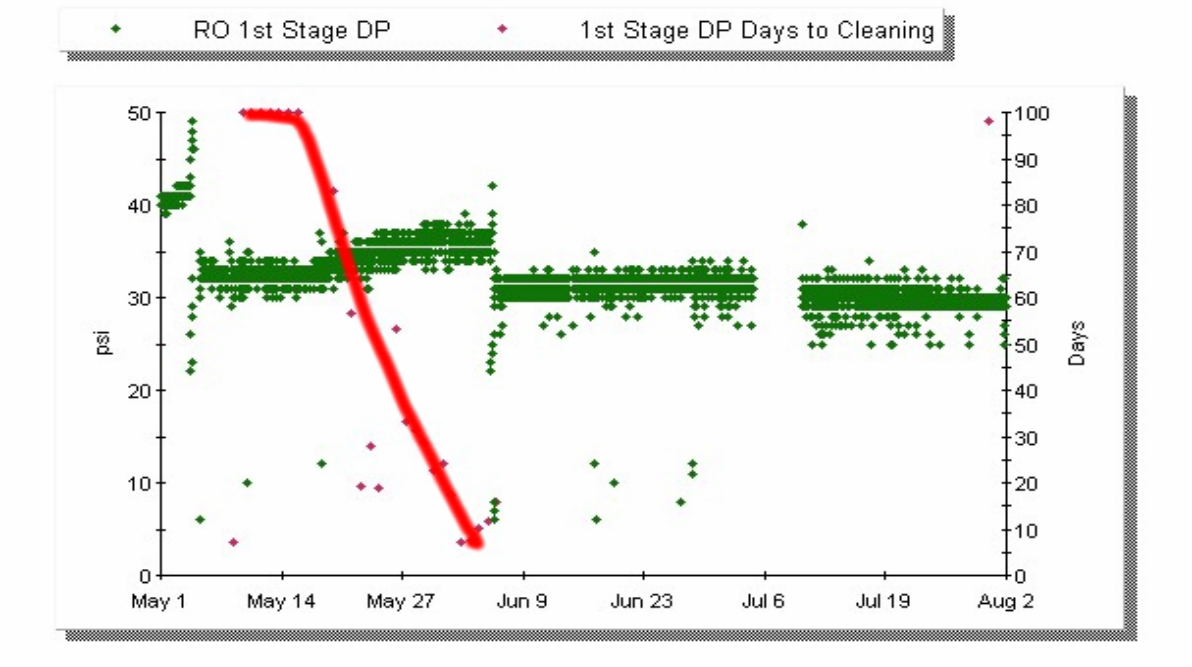


Figure 13.—May-Aug 2000.

The last figure (figure 14) below plots downloaded values in Excel. This allows us to create a bit better and clearer graphs than over the web. The web graph programs balance readability with image size and the resources needed for creation. This still needs some improvement but is not part of the scope of this work.

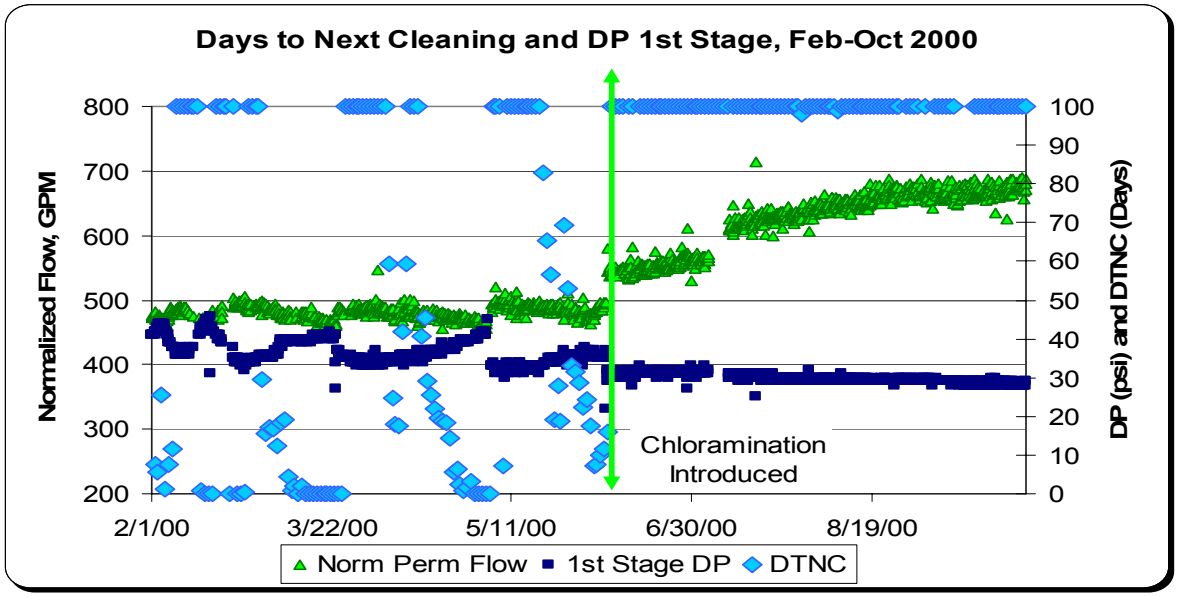


Figure 14.—Feb-Oct 2000.

Once chloramination started, the system performance steadied down and the DTNC values rose to over 100 days. The chloramination process was actually improving the performance as the bio-films decayed and were rinsed out by normal operations. This can be seen by the rising normalized flow values (Norm Perm Flow) and the falling 1st stage DP values. Before chloramination, the system required cleaning at least monthly, as reflected by the rapidly falling DTNC values and actual cleanings which took place.

Commercial Applications

The commercial applications for this refined method are numerous. Not only does this provide a method to handle large quantities of membrane plant data, it offers the promise to be able to do so very quickly. This means that software could potentially replace most of the day to day examination of system performance and greatly reduce the effort needed to oversee longer term operations.

For example, some large districts operating multiple membrane systems in California, contract to outside consultants for review of the membrane plant's operations on a regular basis. The data is downloaded from the DCS systems to Excel spreadsheets and emailed to the consultants. At that office the data is transferred to another Excel program, graphed and trended for personal review. The software developed under this contract, while not completely replacing a human observer, can deliver the data in a ready to go format to a wide range of personnel, greatly reducing the cost of review.

PerLorica already offers the PL-Web / WaterEye service on a commercial basis and has several well known licensees using the system at present. The system monitors both conventional and membrane based plants. Since the membrane information generated by the analysis software modules creates what can be referred to as virtual or computed sensor data, the resulting values can be treated exactly like a real life sensors. Therefore, high and low alarms are set by the end user to trigger and send alerts or alarms as needed.

While the response to calculated data and trends can only take place after the calculation is made, for cleaning and scheduled maintenance the time frames are adequate for almost all systems.

Tables

Table 1.—Raw Data Items

Description	Comments
	NANOFILTRATION SYSTEM
NF 1st Stage Pressure	Same as Feed Pressure
NF 2nd Stage Pressure	2nd stage - 1st Stage = 1st Stage DP
NF Concentrate Conductivity	
NF Concentrate Flow	
NF Concentrate Pressure	Conc Press - 2nd stage = 2nd Stage DP
NF Permeate Conductivity	
NF Permeate Flow	
NF Permeate pH	
NF Permeate Pressure	
NF Raw Water Flow	
NF Raw Water pH	
	REVERSE OSMOSIS
RO 1st Stage Pressure	Same as Feed Pressure
RO 2nd Stage Pressure	2nd stage - 1st Stage = 1st Stage DP
RO Concentrate Conductivity	
RO Concentrate Flow	
RO Concentrate Pressure	Conc Press - 2nd stage = 2nd Stage DP
RO Permeate Conductivity	
RO Permeate Flow	
RO Permeate pH	
RO Permeate Pressure	
RO Raw Water Flow	
RO Raw Water pH	
	SYSTEM and COMMON ITEMS
Raw Water Conductivity	Same as RO / NF feed conductivity
Raw Water pH	
Raw Water Turbidity	Before RO & NF Cartridge and Bag filters
Treated Water Chlorine Residual	Water to the distribution system
Treated Water Conductivity	Water to the distribution system
Treated Water pH	Water to the distribution system
Raw Water Temperature	

Table 2.—Derived Data Items

Description	When Calculated
NANOFILTRATION SYSTEM	
NF 1st Stage DP	On arrival of new data
NF 2nd Stage DP	On arrival of new data
NF Differential Pressure	On arrival of new data
NF Normalized Permeate Flow	On arrival of new data
NF Normalized Salt Passage	On arrival of new data
NF Recovery	On arrival of new data
NF Rejection	On arrival of new data
NF Salt Passage	On arrival of new data
Normalized Flow DTNC	Daily
1st Stage DP Days to Cleaning	Daily
2nd Stage DP Days to Cleaning	Daily
System DP Days to Cleaning	Daily
Salt Passage DTC	Daily
NF Days to Next Cleaning	Daily
BaseLine Reset Functions	Weekly
REVERSE OSMOSIS	
RO 1st Stage DP	On arrival of new data
RO 2nd Stage DP	On arrival of new data
RO Differential Pressure	On arrival of new data
RO Normalized Permeate Flow	On arrival of new data
RO Normalized Salt Passage	On arrival of new data
RO Recovery	On arrival of new data
RO Rejection	On arrival of new data
RO Salt Passage	On arrival of new data
Normalized Flow DTNC	Daily
1st Stage DP Days to Cleaning	Daily
2nd Stage DP Days to Cleaning	Daily
System DP Days to Cleaning	Daily
Salt Passage DTC	Daily
RO Days to Next Cleaning	Daily
BaseLine Reset Functions	Weekly

References

[1] ASTM Methods Designation D4516-00

[2] Byrne, W. *Reverse Osmosis – A Practical Guide for Industrial Users,* Talk Oaks Publishing, pp 218-219, 1995.

[3] Dow Filmtec™ Information taken from the excel spreadsheets (FTNORM) provided by Dow-Filmtec to Port Hueneme for the RO and NF membranes in the system (FT30 and NF-400 respectively).

[4] Dow Filmtec, *Engineering Information* FILMTEC Membranes and DOWEX Ion Exchange Resins - Technical Data, Document 177-01831.PDF, 2003

[5] YSI Incorporated, *"Environmental Monitoring Systems Manual,"* Chapter 5, pp 5-1 to 5-2.

[6] Aquarius Technologies Pty Ltd, Aquarius Technical Bulletin No. 08 – Conductivity Measurements, Queensland, Australia, 2000

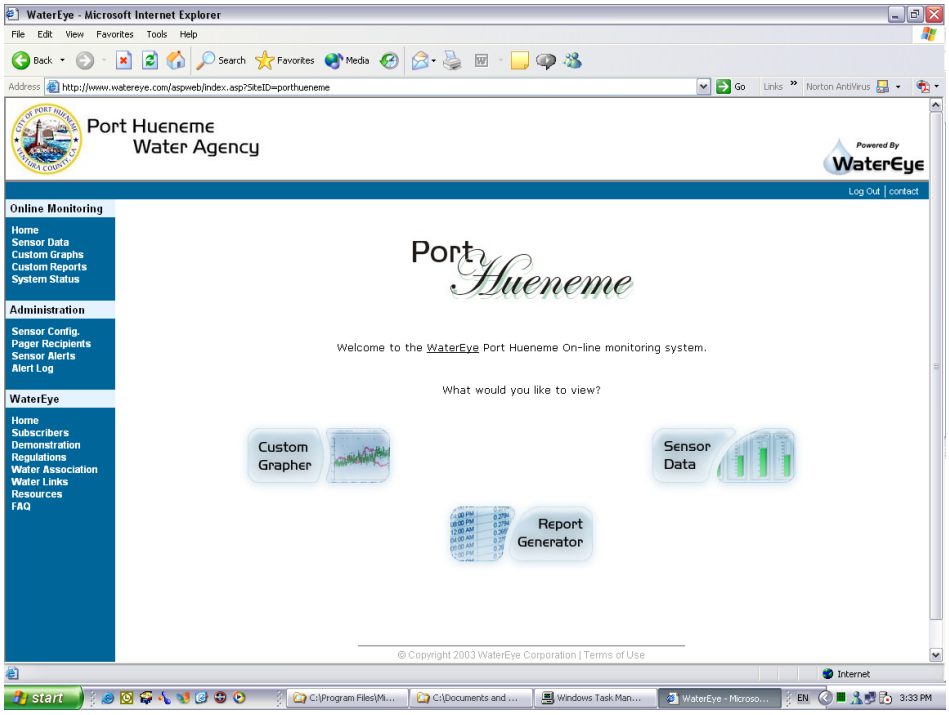
[7] Curve fit using Excel 2000 Standards Data supplied by Oakton Instruments, P.O. Box 5136, Vernon Hills, IL 60061, USA

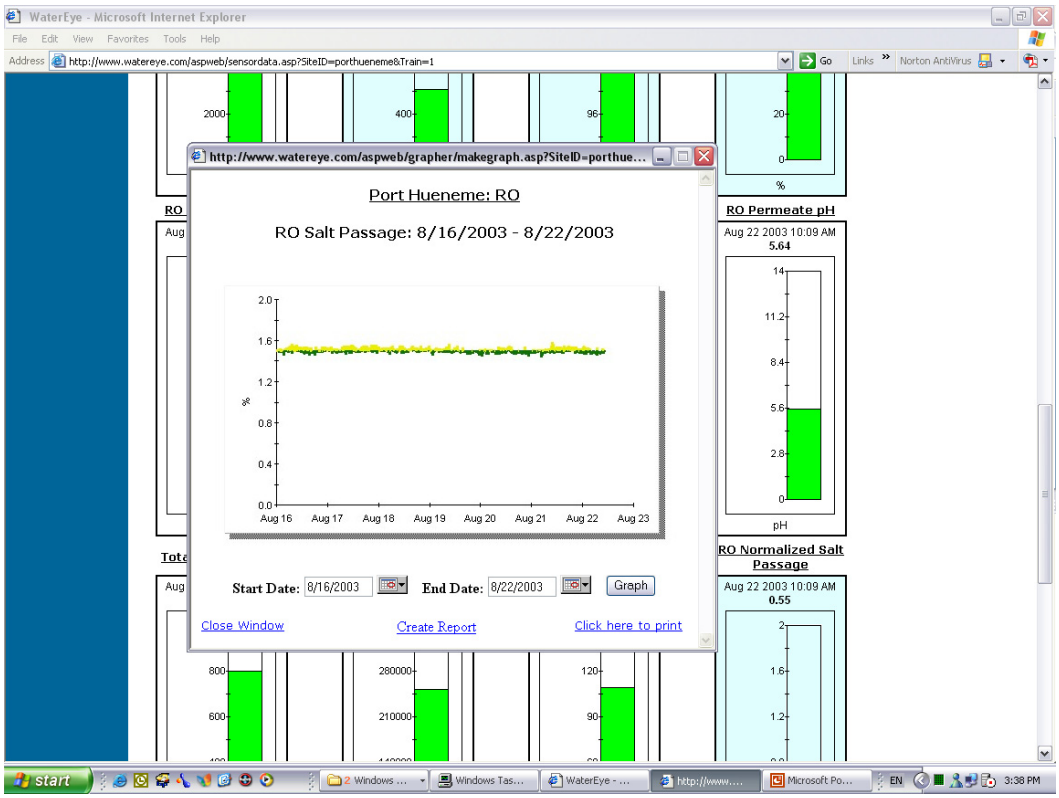
[8] Byrne, *op cit*, p207.

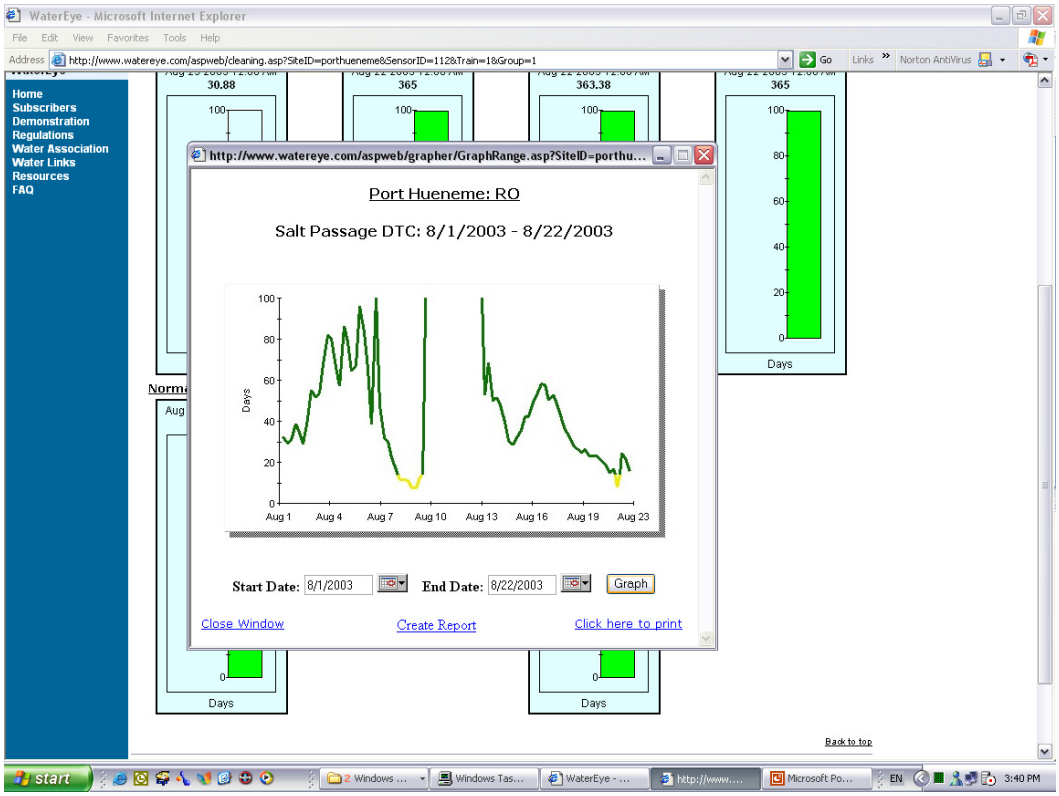
[9] Microsoft Excel, Version 10, Help on Functions. In order of appearance SLOPE(), INTERCEPT(), STDDEV(), RSQ(), 2002.

Appendices

Screen Captures




WaterEye - Microsoft Internet Explorer
http://www.watereye.com/aspweb/sensordata.asp?SiteID=porthueneme&Train=1
http://www.watereye.com/aspweb/grapher/makegraph.asp?SiteID=porthue...
Port Hueneme: RO
RO Salt Passage: 8/16/2003 - 8/22/2003
Aug 16 Aug 17 Aug 18 Aug 19 Aug 20 Aug 21 Aug 22 Aug 23
Start Date: 8/16/2003
End Date: 8/22/2003
Graph
Close Window
Create Report
Click here to print
RO Permeate pH
Aug 22 2003 10:09 AM
5.64
pH
RO Normalized Salt Passage
Aug 22 2003 10:09 AM
0.55


WaterEye - Microsoft Internet Explorer
http://www.watereye.com/aspweb/cleaning.asp?SiteID=porthueneme&SensorID=112&Train=1&Group=1
Home
Subscribers
Demonstration
Regulations
Water Association
Water Links
Resources
FAQ
http://www.watereye.com/aspweb/grapher/GraphRange.asp?SiteID=porthu...
Port Hueneme: RO
Salt Passage DTC: 8/1/2003 - 8/22/2003
Aug 1 Aug 4 Aug 7 Aug 10 Aug 13 Aug 16 Aug 19 Aug 23
Start Date: 8/1/2003
End Date: 8/22/2003
Graph
Close Window
Create Report
Click here to print
Days
Back to top

Visual Basic Code

```
Option Explicit
    Dim adoRS As Recordset      ' Our recordset
    Dim xs As Variant           ' Array of x values
    Dim ys As Variant           ' Array of y values
    Dim m_clsLinReg(0 To 2) As clsLinearRegression 'holds all the Linear Regression info for each
look back period
    'These variables are only needed when data does not exist on the lookback date
    Dim offset1 As Integer      ' #of days from MaxLB until there is data in the recordset
(usually 0)
    Dim offset2 As Integer      ' #of days from MidLB until there is data in the recordset
(usually 0)
    Dim offset3 As Integer      ' #of days from MinLB until there is data in the recordset
(usually 0)

    Dim StartDate As Date       ' Day to start calculating days to next cleaning from (usually =
to today)

' These can also come from the database. Use set values for
' development purposes.
Const MAXLB = 30
Const MIDLB = 20
Const MINLB = 7
Const SIGMA = 1
Const CONFIDENCE = 0.8
Const MAXDAYS = 365
---------------------------------------

'Calculate Days to Next Cleaning
Sub CalcDaysToClean(DataTable As String, SensorTable As String, ConstantsTable As String, ByRef
db As Connection, TrainCount As Integer, dStartDate As Date)

    Dim i As Integer    'control variable in train loop
    Dim j As Integer    'control variable in sensor loop

    Dim SensorCount As Integer      ' the number of sensors days to next cleaning will be
calculated for
    Dim Sensor(1 To 5) As Integer   ' the list of sensors days to next cleaning will be
calculated for

    Dim Days As Double              ' Number of days until next cleaning (the smallest value
returned from GetDaysToNextCleaning)
    Dim tmpDays As Double           ' The current days returned from GetDaysToNextCleaning

On Error GoTo ERR_Handler

    StartDate = dStartDate

    ' For each Train calculate the days to next cleaning
    For i = 1 To TrainCount
        SensorCount = 0
        Days = MAXDAYS
        'check to see if we are calculating days to next cleaning
        If (SensorExists(SensorTable, 112, i, db)) Then

            'check to see what sensors we will calculate days to next cleaning on
            If (SensorExists(SensorTable, 170, i, db)) Then
                SensorCount = SensorCount + 1
                Sensor(SensorCount) = 170
            End If
            If (SensorExists(SensorTable, 171, i, db)) Then
                SensorCount = SensorCount + 1
                Sensor(SensorCount) = 171
            End If
            If (SensorExists(SensorTable, 172, i, db)) Then
                SensorCount = SensorCount + 1
                Sensor(SensorCount) = 172
```

```
            End If
            If (SensorExists(SensorTable, 173, i, db)) Then
                SensorCount = SensorCount + 1
                Sensor(SensorCount) = 173
            End If
            If (SensorExists(SensorTable, 174, i, db)) Then
                SensorCount = SensorCount + 1
                Sensor(SensorCount) = 174
            End If

            If (SensorCount > 0) Then
                For j = 1 To SensorCount
                    'get days to next cleaning for current sensors
                    If (Sensor(j) = 173) Then
                        tmpDays = GetDaysToNextCleaning(DataTable, SensorTable, ConstantsTable,
db, Sensor(j), i, True)
                    Else
                        tmpDays = GetDaysToNextCleaning(DataTable, SensorTable, ConstantsTable,
db, Sensor(j), i, False)
                    End If
                    If tmpDays > MAXDAYS Then tmpDays = MAXDAYS
                    If (tmpDays <= MAXDAYS And tmpDays <> -666) Then
                        If (tmpDays < 0) Then tmpDays = 0
                        'insert into SQL sensor cleaning value
                        SQLUpdate.InsertData Sensor(j), i, StartDate, tmpDays, db, DataTable
                    End If
                    If (tmpDays < Days And tmpDays <> -666) Then Days = tmpDays
                Next j
                If Days <= MAXDAYS Then
                    'insert into SQL system cleaning value (the lowest of the sensor cleaning
values)
                    SQLUpdate.InsertData 112, i, StartDate, Days, db, DataTable
                End If
            Else
                WriteLog ("No Sensor Existed in " & SensorTable & " to base days to next cleaning
on.")
            End If
        Else
            WriteLog ("We are not calculating days to next cleaning for train " & i)
        End If
    Next i

Exit Sub

ERR_Handler:
    WriteLog "CalcDaysToClean Error " & Err.Number & " - " & Err.Description
End Sub
---------------------------------------

Function GetDaysToNextCleaning(DataTable As String, SensorTable As String, ConstantsTable As
String, ByRef db As Connection, ByVal SensorID As Integer, Train As Integer, Flipped As Boolean)
As Double
    Dim adoDataRS As Recordset
    Dim strSQL As String

    Dim Days1 As Double     ' Days to Next Cleaning based on the Maximum # of lookback days
    Dim Days2 As Double     ' Days to Next Cleaning based on the Middle # of lookback days
    Dim Days3 As Double     ' Days to Next Cleaning based on the Minimum # of lookback days

    Dim SensorIDb As Integer    'The sensor that the current Days To Cleaning sensor is based on
                                '(i.e. Sensor 170 (1st Stg DP Cleaning) is based on 144 (1st Stg
DP)

    Dim CleanLevel As Double    'The value at which we need to do a cleaning for the current
sensor

On Error GoTo ERR_Handler


    ' Get Constants from constants table in SQL
    Set adoDataRS = New Recordset
```

```
    strSQL = "SELECT * from " & ConstantsTable & " WHERE TrainNumber='" & Train & "' AND
SensorID='" & SensorID & "'"
    adoDataRS.Open strSQL, db, adOpenStatic, adLockReadOnly

    If (Not adoDataRS.EOF And Not adoDataRS.BOF) Then
        SensorIDb = adoDataRS!Constant
        CleanLevel = adoDataRS!Value * adoDataRS!Factor
    End If

    adoDataRS.Close
    Set adoDataRS = Nothing

    Set m_clsLinReg(0) = New clsLinearRegression
    Set m_clsLinReg(1) = New clsLinearRegression
    Set m_clsLinReg(2) = New clsLinearRegression

 ' Get the first pass Linear Regression Info (ignoring values <=0) for each look back period
    Get1stPassInfo SensorIDb, Train, DataTable, db, 0
    m_clsLinReg(0).GetSlope
    m_clsLinReg(0).GetStDev
    m_clsLinReg(0).GetYIntercept
    m_clsLinReg(0).GetRSQ
    DoEvents
    Get1stPassInfo SensorIDb, Train, DataTable, db, 1
    m_clsLinReg(1).GetSlope
    m_clsLinReg(1).GetStDev
    m_clsLinReg(1).GetYIntercept
    m_clsLinReg(1).GetRSQ
    DoEvents
    Get1stPassInfo SensorIDb, Train, DataTable, db, 2
    m_clsLinReg(2).GetSlope
    m_clsLinReg(2).GetStDev
    m_clsLinReg(2).GetYIntercept
    m_clsLinReg(2).GetRSQ
    DoEvents

    ' Get the data again, this time only getting data w/i a specified sigma (1 sigma = 1 X StDev)
of the average
    ' If after the first pass the standard deviation is very small or the confidence very high,
don't try to get the data w/i a specified sigma (you won't get good data if any)
    If (m_clsLinReg(0).GetStDev() / m_clsLinReg(0).arrayMean(m_clsLinReg(0).ys) > 0.03 And
m_clsLinReg(0).GetRSQ() < 0.9) Then Get2ndPassInfo SensorIDb, Train, DataTable, db, 0
    m_clsLinReg(0).GetSlope
    m_clsLinReg(0).GetStDev
    m_clsLinReg(0).GetYIntercept
    m_clsLinReg(0).GetRSQ
    DoEvents
    ' If after the first pass the standard deviation is very small or the confidence very high,
don't try to get the data w/i a specified sigma (you won't get good data if any)
    If (m_clsLinReg(1).GetStDev() / m_clsLinReg(1).arrayMean(m_clsLinReg(1).ys) > 0.03 And
m_clsLinReg(1).GetRSQ() < 0.9) Then Get2ndPassInfo SensorIDb, Train, DataTable, db, 1
    m_clsLinReg(1).GetSlope
    m_clsLinReg(1).GetStDev
    m_clsLinReg(1).GetYIntercept
    m_clsLinReg(1).GetRSQ
    DoEvents
    ' If after the first pass the standard deviation is very small or the confidence very high,
don't try to get the data w/i a specified sigma (you won't get good data if any)
    If (m_clsLinReg(2).GetStDev() / m_clsLinReg(2).arrayMean(m_clsLinReg(2).ys) > 0.03 And
m_clsLinReg(2).GetRSQ() < 0.9) Then Get2ndPassInfo SensorIDb, Train, DataTable, db, 2
    m_clsLinReg(2).GetSlope
    m_clsLinReg(2).GetStDev
    m_clsLinReg(2).GetYIntercept
    m_clsLinReg(2).GetRSQ
    DoEvents

    'Do predictions based on the 3 lookback periods
    Days1 = Format(DaysToNextCleaning(m_clsLinReg(0).GetYIntercept, m_clsLinReg(0).GetSlope,
CleanLevel, MAXLB - offset1, Flipped), "0.###")
    Days2 = Format(DaysToNextCleaning(m_clsLinReg(1).GetYIntercept, m_clsLinReg(1).GetSlope,
CleanLevel, MIDLB - offset2, Flipped), "0.###")
```

```
    Days3 = Format(DaysToNextCleaning(m_clsLinReg(2).GetYIntercept, m_clsLinReg(2).GetSlope,
CleanLevel, MINLB - offset3, Flipped), "0.###")
    '****************************************
    ' START PREDICTION ANALYSIS
    '****************************************
    ' Determine which prediction to use
    Dim dMin As Double  ' The fewest days until days to next cleaning
    Dim n As Integer    ' Which LB prediction to use
    dMin = MAXDAYS      ' This should be the max # of days to next cleaning, or the number of
days to the scheduled cleaning

    ' check to see if we have accurate results (R > X OR StDev is very small... but > than 0)
    If (((m_clsLinReg(0).GetRSQ() > CONFIDENCE And (m_clsLinReg(0).GetStDev() / CleanLevel <
0.15)) Or (m_clsLinReg(0).GetStDev() / m_clsLinReg(0).arrayMean(m_clsLinReg(0).ys)) <= 0.03) And
m_clsLinReg(0).GetStDev() > 0) Then
        ' If this prediction is smaller than the current fewest days to cleaning
        If (Days1 < dMin) Then
            dMin = Days1 ' update the fewest days to next cleaning
            n = 1           ' set the LB prediction being used
        Else
            n = 1
        End If
    End If
    If (((m_clsLinReg(1).GetRSQ() > CONFIDENCE And (m_clsLinReg(1).GetStDev() / CleanLevel <
0.15)) Or (m_clsLinReg(1).GetStDev() / m_clsLinReg(1).arrayMean(m_clsLinReg(1).ys)) <= 0.03) And
m_clsLinReg(1).GetStDev() > 0) Then
        If (Days2 < dMin) Then
            dMin = Days2
            n = 2
        Else
            If (n = 0) Then n = 2
        End If
    End If
    If (((m_clsLinReg(2).GetRSQ() > CONFIDENCE And (m_clsLinReg(2).GetStDev() / CleanLevel <
0.15)) Or (m_clsLinReg(2).GetStDev() / m_clsLinReg(2).arrayMean(m_clsLinReg(2).ys)) <= 0.03 And
m_clsLinReg(2).GetStDev() > 0)) Then
        If (Days3 < dMin) Then
            dMin = Days3
            n = 3
        Else
            If (n = 0) Then n = 3
        End If
    End If

    ' if we couldn't predict anything with X confidence, cut the confidence requirement in half
and try again
    If (n = 0) Then
        ' check to see if we have accurate results (R > X OR StDev is very small... but > than 0)
        If (((m_clsLinReg(0).GetRSQ() > CONFIDENCE / 2 And (m_clsLinReg(0).GetStDev() /
CleanLevel < 0.15)) Or (m_clsLinReg(0).GetStDev() / m_clsLinReg(0).arrayMean(m_clsLinReg(0).ys))
<= 0.03) And m_clsLinReg(0).GetStDev() > 0) Then
            If (Days1 < dMin) Then
                dMin = Days1
                n = 1
            Else
                n = 1
            End If
        End If
        If (((m_clsLinReg(1).GetRSQ() > CONFIDENCE / 2 And (m_clsLinReg(1).GetStDev() /
CleanLevel < 0.15)) Or (m_clsLinReg(1).GetStDev() / m_clsLinReg(1).arrayMean(m_clsLinReg(1).ys))
<= 0.03) And m_clsLinReg(1).GetStDev() > 0) Then
            If (Days2 < dMin) Then
                dMin = Days2
                n = 2
            Else
                If (n = 0) Then n = 2
            End If
        End If
        If (((m_clsLinReg(2).GetRSQ() > CONFIDENCE / 2 And (m_clsLinReg(2).GetStDev() /
CleanLevel < 0.15)) Or (m_clsLinReg(2).GetStDev() / m_clsLinReg(2).arrayMean(m_clsLinReg(2).ys))
<= 0.03 And m_clsLinReg(2).GetStDev() > 0)) Then
```

```
            If (Days3 < dMin) Then
                dMin = Days3
                n = 3
            Else
                If (n = 0) Then n = 3
            End If
        End If
    End If

    'If we were able to make a prediction
    If (n > 0) Then
        Select Case (n)
        Case 1:
            GetDaysToNextCleaning = Days1
        Case 2:
            GetDaysToNextCleaning = Days2
        Case 3:
            GetDaysToNextCleaning = Days3
        End Select

        If (GetTodaysAverage(SensorIDb, Train, DataTable, db) > CleanLevel) Then
            WriteLog ("Time for a cleaning.")
        End If
    'Not enough confidence to make prediction
    Else
        GetDaysToNextCleaning = -666
        WriteLog ("Unable to make prediction.")
    End If
    '****************************************
    ' END PREDICTION ANALYSIS
    '****************************************

    Set m_clsLinReg(0) = Nothing
    Set m_clsLinReg(1) = Nothing
    Set m_clsLinReg(2) = Nothing


Exit Function
ERR_Handler:
    WriteLog "GetDaysToNextCleaning Error " & Err.Number & " - " & Err.Description
    On Error Resume Next
    GetDaysToNextCleaning = -666
    Set m_clsLinReg(0) = Nothing
    Set m_clsLinReg(1) = Nothing
    Set m_clsLinReg(2) = Nothing
End Function
---------------------------------------

Private Sub Get1stPassInfo(SensorID As Integer, Train As Integer, DataTable As String, db As
Connection, LB As Integer)
    Dim LBDate As Date
    Dim nLBDays As Integer

On Error GoTo ERR_Handler

    'find out how many days to look back
    Select Case LB
        Case 0:
            nLBDays = MAXLB
        Case 1:
            nLBDays = MIDLB
        Case 2:
            nLBDays = MINLB
    End Select

    'get the data (> than 0) for the current lookback period
    Set adoRS = GetData(SensorID, Train, DataTable, db, nLBDays, 0)

    'check to see if we have data
    If Not adoRS Is Nothing Then
        Dim offset As Integer
```

```
        LBDate = StartDate - nLBDays
        'take the data from the record set and put it into two arrays
        If modAssist.FillArrays(LBDate, adoRS, xs, ys, offset) Then
            'handle situation when there is no data at the LB date
            Select Case LB
                Case 0:
                    offset1 = offset
                Case 1:
                    offset2 = offset
                Case 2:
                    offset3 = offset
            End Select

            m_clsLinReg(LB).xs = xs
            m_clsLinReg(LB).ys = ys

            'calculate the linear regression info
            m_clsLinReg(LB).GetSlope
            m_clsLinReg(LB).GetStDev
            m_clsLinReg(LB).GetRSQ
            m_clsLinReg(LB).GetYIntercept
        End If
        adoRS.Close
    End If

Exit Sub

ERR_Handler:
    WriteLog "Sub Get1stPassInfo " & Err.Number & " - " & Err.Description
End Sub
---------------------------------------

Private Sub Get2ndPassInfo(SensorID As Integer, Train As Integer, DataTable As String, db As
Connection, LB As Integer)
    Dim dSigma As Double    ' the number of sigmas to ignore above and below the average
    Dim dAve As Double      ' the average of the data
    Dim nLBDays As Integer  ' the number of look back days
    Dim LBDate As Date      ' the start of the look back period

On Error GoTo ERR_Handler

    'find out how many days to look back
    Select Case LB
        Case 0:
            nLBDays = MAXLB
        Case 1:
            nLBDays = MIDLB
        Case 2:
            nLBDays = MINLB
    End Select

    dSigma = SIGMA
    dAve = m_clsLinReg(LB).arrayMean(m_clsLinReg(LB).ys)

    'get the data (ignoring data < and > nSigmas from the average) for the current lookback
period
    If (dSigma <> 0) Then
        Set adoRS = GetData(SensorID, Train, DataTable, db, nLBDays, dAve - (dSigma *
m_clsLinReg(LB).GetStDev), dAve + (dSigma * m_clsLinReg(LB).GetStDev))

        On Error Resume Next
        'check to see if we have enough data (we need data for at least 1/3 of the days in the
look back period)
        If ((adoRS Is Nothing) Or adoRS.RecordCount < (nLBDays / 3)) Then
            Dim nRecs As Integer

            If (adoRS Is Nothing) Then
                WriteLog ("Warning: We have data for 0 out of a possible " & nLBDays & " days.")
                nRecs = 0
            Else
```

```
                WriteLog ("Warning: We have data for " & adoRS.RecordCount & " out of a possible
" & nLBDays & " days.")
                nRecs = adoRS.RecordCount
            End If

            'Try another method of getting data
            'Get data w/i 10% of the average
            adoRS.Close
            Set adoRS = GetData(SensorID, Train, DataTable, db, nLBDays, dAve * (0.9), dAve *
(1.1))

            If (adoRS Is Nothing) Then
                'This method of getting data resulted in 0 records
                WriteLog ("Warning: No data when getting data w/i 10% of the average")
                Set adoRS = GetData(SensorID, Train, DataTable, db, nLBDays, dAve - (dSigma *
m_clsLinReg(LB).GetStDev), dAve + (dSigma * m_clsLinReg(LB).GetStDev))
            Else
                'We got some data, now see if it's more than what we had before
                WriteLog ("Info: Got " & adoRS.RecordCount & " days of data when getting data w/i
10% of the average")
                If (adoRS.RecordCount < nRecs) Then
                    Set adoRS = GetData(SensorID, Train, DataTable, db, nLBDays, dAve - (dSigma *
m_clsLinReg(LB).GetStDev), dAve + (dSigma * m_clsLinReg(LB).GetStDev))
                End If
            End If

        End If
        On Error GoTo ERR_Handler
    Else
        Set adoRS = GetData(SensorID, Train, DataTable, db, nLBDays, 0)
    End If

    'remove the data from the 1st pass
    m_clsLinReg(LB).Clear

    'check to see if we have data
    If Not adoRS Is Nothing Then
        Dim offset As Integer
        LBDate = StartDate - nLBDays
        'take the data from the recordset and put it into two arrays
        If modAssist.FillArrays(LBDate, adoRS, xs, ys, offset) Then

            'handle situation when there is no data at the LB period
            Select Case LB
                Case 0:
                    offset1 = offset
                Case 1:
                    offset2 = offset
                Case 2:
                    offset3 = offset
            End Select

            m_clsLinReg(LB).xs = xs
            m_clsLinReg(LB).ys = ys

            'calculate the linear regression info
            m_clsLinReg(LB).GetSlope
            m_clsLinReg(LB).GetStDev
            m_clsLinReg(LB).GetRSQ
            m_clsLinReg(LB).GetYIntercept
        End If
        adoRS.Close
    End If

Exit Sub

ERR_Handler:
    WriteLog "Sub Get2ndPassInfo " & Err.Number & " - " & Err.Description
End Sub
---------------------------------------
```

```
'Gets the average data for today (ignoring values < 0. Used to determine if we are above the
cleaning value.
Public Function GetTodaysAverage(SensorID As Integer, Train As Integer, DataTable As String, db
As Connection) As Double

On Error GoTo ERR_Handler

    Set adoRS = GetData(SensorID, Train, DataTable, db, 1, 0)
    If Not adoRS Is Nothing Then
        If ((Not adoRS.EOF) And (Not adoRS.BOF)) Then
            GetTodaysAverage = adoRS!Data
            adoRS.Close
        Else
            GetTodaysAverage = 0
        End If
    End If

Exit Function

ERR_Handler:
    WriteLog "Sub GetTodaysAverage " & Err.Number & " - " & Err.Description
End Function
---------------------------------------

'Returns a recordset of the data during the lookback period between the LowData and HighData
Function GetData(SensorID As Integer, Train As Integer, DataTable As String, db, LookBack As
Integer, LowData As Double, Optional HighData As Double = -1) As Recordset
    Dim strSQL As String
On Error GoTo ERR_Handler

    Set adoRS = New Recordset

    'if there was no input for HighData get all data > LowData
    If (HighData = -1) Then
        strSQL = "SELECT TimeDate = convert(varchar, TimeDate, 101), Data = Avg(Data) FROM " &
DataTable & " WHERE Data>'" & LowData & "' AND TrainNumber='" & Train & "' AND SensorType='" &
SensorID & "' And TimeDate>'" & StartDate - LookBack & "' and TimeDate<='" & StartDate & "' GROUP
BY convert(varchar, TimeDate, 101) Order By convert(varchar, TimeDate, 101)"
    'if there was a value for HighData get all data > LowData and < HighData
    Else
        strSQL = "SELECT TimeDate = convert(varchar, TimeDate, 101), Data = Avg(Data) FROM " &
DataTable & " WHERE Data>'" & LowData & "' AND Data<'" & HighData & "' AND TrainNumber='" & Train
& "' AND SensorType='" & SensorID & "' And TimeDate>'" & StartDate - LookBack & "' and
TimeDate<='" & StartDate & "' GROUP BY convert(varchar, TimeDate, 101) Order By convert(varchar,
TimeDate, 101)"
    End If
    adoRS.Open strSQL, db, adOpenStatic, adLockReadOnly

    If (Not adoRS.EOF And Not adoRS.BOF) Then
       Set GetData = adoRS
    Else
        WriteLog ("No Data found for Sensor " & SensorID)
        adoRS.Close
        Set adoRS = Nothing
        Set GetData = Nothing
    End If

Exit Function

ERR_Handler:
    WriteLog "Function GetData " & Err.Number & " - " & Err.Description
    On Error Resume Next
    Set GetData = Nothing
    If (Not adoRS Is Nothing) Then
        adoRS.Close
        Set adoRS = Nothing
    End If
End Function
---------------------------------------
```

```
Function DaysToNextCleaning(YIntercept As Double, Slope As Double, CleanValue As Double, LBPeriod
As Integer, Flipped As Boolean) As Double

    'If we expect a negative slope to determine DTNC
    If Flipped Then
        If (Slope >= 0) Then
            DaysToNextCleaning = MAXDAYS 'This should be the maximum that days to next cleaning
will be, or the next scheduled cleaning
        Else
            DaysToNextCleaning = (CleanValue - YIntercept) / Slope - LBPeriod
        End If
    Else
        If (Slope <= 0) Then
            DaysToNextCleaning = MAXDAYS 'This should be the maximum that days to next cleaning
will be, or the next scheduled cleaning
        Else
            DaysToNextCleaning = (CleanValue - YIntercept) / Slope - LBPeriod
        End If
    End If

End Function
---------------------------------------

Function SensorExists(SensorTable As String, SensorID As Integer, ByVal Train As String, ByRef db
As Connection) As Boolean
    Dim adoDataRS As Recordset
    Dim strSQL As String

On Error GoTo ERR_Handler

    Set adoDataRS = New Recordset
    strSQL = "SELECT SensorType FROM " & SensorTable & " WHERE SensorType='" & SensorID & "' AND
TrainNumber='" & Train & "'"
    adoDataRS.Open strSQL, db, adOpenStatic, adLockReadOnly

    If (Not adoDataRS.EOF And Not adoDataRS.BOF) Then
        SensorExists = True
    Else
        SensorExists = False
    End If

    adoDataRS.Close
    Set adoDataRS = Nothing


Exit Function
ERR_Handler:
    SensorExists = False
    adoDataRS.Close
    Set adoDataRS = Nothing
    WriteLog "SensorExists Error " & Err.Number & " - " & Err.Description

End Function
---------------------------------------
Sub WriteLog(Msg As String)
    Start.WriteLog (Msg)
End Sub

---------------------------------------

Function GetNewBaseline(Train As Integer, DataTable As String, CleaningSensor As Integer, curBL
As Double) As Double
    Dim SensorID As Integer
    Dim newBL As Double
    Dim stdDev As Double

On Error GoTo ERR_Handler

    GetNewBaseline = 0

    Select Case (CleaningSensor)
```

```
        Case 170
            SensorID = 144
        Case 171
            SensorID = 145
        Case 172
            SensorID = 8
        Case 173
            SensorID = 14
        Case 174
            SensorID = 110
    End Select

    'gets the data from sql ignoring data when system offline (need at least 25 values)
    If (RetrieveSQLData(Train, SensorID, SensorTable, DataTable, curBL) > 25) Then
        m_clsLinReg.xs = xs
        m_clsLinReg.ys = ys

        newBL = m_clsLinReg.arrayMean(m_clsLinReg.ys)
        stdDev = m_clsLinReg.GetStDev()
        If (stdDev / newBL) < 0.0233 Then
            GetNewBaseline = newBL
        End If
    End If
    DoEvents
    m_clsLinReg.Clear

Exit Function
ERR_Handler:
    GetNewBaseline = 0
    WriteLog "GetCurrentBaseline Error " & Err.Number & " - " & Err.Description
End Function
```