DATA CONSIDERATIONS FOR SCADA PLANNING AND SETUP

Robert J. Strand[1]
Albert J. Clemmens[2]

ABSTRACT

The setup of a new SCADA system can be a daunting task. In addition to sifting through the myriad of sensors, programmable controllers, communication infrastructures, and SCADA software packages available in today's market, managers have to determine what they are going to monitor or control, what information they need to acquire from the field, what information will be sent out to field sites, the formats used to store and convey that information, and the need to archive select portions of that data for historical purposes. Even if a new implementation is done in phases, some forethought and advance preparation can simplify the process. The recommendations presented are based on experience gained from over 12 years of SCADA implementation associated with canal automation research at the U.S. Arid-Land Agricultural Research Center (ALARC) in Maricopa, AZ.

INTRODUCTION

The ALARC has been involved in canal automation research for over a decade. Beginning in 1995, the center (then known as the U.S. Water Conservation Laboratory in Phoenix, AZ) has been utilizing the WM-lateral of the Maricopa Stanfield Irrigation and Drainage District (MSIDD) in Maricopa as a platform for conducting automatic canal control research. Since 2002, the ALARC has also been assisting the Central Arizona Irrigation and Drainage District (CAIDD) of Eloy, AZ in the implementation of a supervisory control system. At present, CAIDD is able to monitor and control 121 telemetry sites on its distribution network. Recently, CAIDD has outfitted sites on three of their lateral canals with additional hardware to begin a move toward district-wide automatic control. Preliminary testing of the automatic control was conducted in the fall of 2006 and full testing will commence in the summer of 2007.

This paper is an overview of the experiences of the ALARC staff and their cooperators. It is not intended to be a SCADA implementation manual, but rather a list of considerations that may prove useful in the planning, development, and

[1] Electrical Engineer, USDA-ARS, U.S. Arid-Land Agricultural Research Center., 21881 N. Cardon Ln., Maricopa, AZ 85239; bstrand@uswcl.ars.ag.gov
[2] Research Leader and Research Hydraulic Engineer, USDA-ARS, U.S. Arid-Land Agricultural Research Center., 21881 N. Cardon Ln., Maricopa, AZ 85239; bclemmens@uswcl.ars.ag.gov

deployment of a flexible and functional SCADA system. It is intended that this information not be specific to any particular hardware or software manufacturer.

GENERAL CONSIDERATIONS

Bits, Bytes, Integers, and Real Numbers

Digital computers handle data using a discrete component called a "bit"; short for **bi**nary digi**t**. In our standard decimal system, each digit can be assigned one of ten values (0 to 9). Similarly, in the binary system each bit can be assigned one of two values (0 or 1). While there are digital processor functions that will allow one to manipulate a single bit of information, processors commonly access binary information in groups of 8-bits, commonly called a byte, and larger groupings in the form of 16, 32, or 64-bit "words".

In their native setting, digital processors generally use these groupings to represent integer values, i.e. 0,1,2,3, etc. and the processors are very efficient in executing integer arithmetic. In an unsigned format, the range of a binary number is from 0 to 2^n-1, where n represents the number of digits in the number. So, an unsigned byte can have a value from 0 to 255. Signed integer values are generally represented using what is called the Two's Complement format, giving a range of $-(2^{(n-1)})$ to $2^{(n-1)}$-1 (-128 to 127) for a byte.

We tend to think of numerical data in terms of a continuous spectrum of real numbers. When typing the value of 9.81 into a spreadsheet to represent the acceleration of gravity in SI units, we give little thought as to how the computer actually stores this information. The value of 9.81 is stored in word groupings using what is called a floating-point format; meaning that the decimal point (more appropriately called the radix point for systems such as binary) can float from one instance to another. The floating-point format breaks the continuous range of real numbers into a set of discrete values bounded on the positive and negative ends. Generally, this isn't a problem in our everyday use of a personal computer because responsible programmers utilize a version of the floating-point format, called 64-bit or double-precision. This version of the format has a fine discritization and the bounds allow for very large positive and negative numbers. There is also a single-precision (32-bit) version of the floating-point format which has a more coarse discritization and narrower range.

In the average personal computer, floating-point arithmetic operations are implemented in a separate math co-processor that resides on the same physical chip as the microprocessor. In older machines, such as those using Intel processors older than the Intel 80486, this processor was a separate optional chip. For these older systems, developers were forced to include floating-point emulators in their software. While these emulators were slow, they enabled program execution on machines without the processor.

Real number arithmetic can also be implemented on systems without floating-point capabilities through the fixed-point representation; so named because all instances have the same number of digits to the left and right of the radix point. In the fixed-point representation, values that would normally be floating-point are scaled by a power of two, stored as integers, manipulated using standard integer arithmetic, and then rescaled when they need to be displayed or printed.

While these numeric representations really shouldn't be a concern in the general use of a modern computer, they can be a concern in a SCADA implementation when considering the capabilities of field hardware and the communication protocols used to convey information between field devices and the central SCADA software.

The numerical capability of field hardware ranges from full double-precision floating-point to only simple integer functionality. The trade-off is one of flexibility and ease of use versus cost. The hardware that does not support floating-point is usually less expensive, but it may require some custom programming if a need arises for the field unit to use storage or calculations in a floating or fixed-point format.

Many protocols, including Modbus (with quasi-standard extensions), are compatible with varying bit-widths of the floating-point format as well as signed/unsigned integer formats. Issues can arise with communication protocols when the field unit is programmed to use custom fixed-point representations for values, such as a flow or upstream level setpoint, that need to be transmitted between the field unit and the SCADA software. Some customization may be needed in the SCADA software to be able to properly communicate information to the remote sites in the fixed-point format.

A-D Resolution

In order for a digital device such as a PLC to interact with analog devices like a pressure transducer, the PLC utilizes a process called Analog-to-Digital (A-D) conversion to convert a sensor output signal, e.g. electrical current, into a discrete digital value that can be utilized by the PLC. A-D converters are specified in varying bit-widths. One can find devices with A-D conversion on the order of 8, 10, 12, and 16-bits. Generally, 8-bit resolution is less expensive but in practice, the measurement can be too coarse. For instance, if used with a gate position sensor with a range of 4 ft, the reading of that transducer would be broken into 255 increments of approximately 0.2 inches. A finer resolution, such as 16-bit, can be a problem as well as it can result in a noisy signal. Applied to the prior example, a 16-bit A-D converter would break the gate position reading into 65535 increments each representing about 0.0007 inches. The 10 and 12-bit implementations seem to work best.

DATA CONSIDERATIONS

Physical Information

The physical dimensions and design information of various structures may be needed in a SCADA implementation. This information can be used to determine operating criteria, such as seasonal operating depths, and used in conjunction with measured data to compute useful management data such as flow rates at regulating structures. Some structures and examples of associated useful information are summarized in Table 1.

Table 1. Physical Dimensions and Design Information

Structure	Dimensions and Design Information
Sluice Gate	Gate Height Gate Width Sill Elevation
Radial Gate	Gate Radius Gate Width Trunion (Pinion) Height Sill Elevation
Weirs (Sharp Crested)	Crest Elevation Crest Length
Canal Sections	Design Flow Capacity and Water Levels Cross-Section Information
Pipe Sections	Pipe Diameter and Length

Depending on the distribution of functionality throughout the SCADA system, this information may be used in the field hardware as well as in software on the central SCADA system. If so, storing this data in a computerized database may be advantageous for a number of reasons:

(1) Many SCADA software packages are capable of reading information from standard database formats and using it to populate internal data structures.
(2) Should this information be utilized by field units for local calculations, it would be possible to use the same database with a configuration program; automatically converting to the appropriate numerical system as needed.
(3) By minimizing the number of storage locations, data entry mistakes can be minimized.

In addition to the dimensions of the physical structures, it may be advantageous to document the topology, i.e. the spatial or upstream/downstream relationship of the regulating structures, and the locations of supply and demand points in the system. With this information, it would be possible to incorporate supply/demand

information into the SCADA system and provide operators with the local and downstream demands for a given structure.

Sensor Calibrations

While sensor calibrations shouldn't have to be done often (seasonally or quarterly), periodic field calibrations are necessary in order to give operators an accurate depiction of the field situation. Many sensors, including pressure transducers for water levels, potentiometers used for vertical gate position, and inclinometers used for radial gate position can be considered linear devices. The ALARC has adopted a linear calibration method that applies to each of these sensors.

$$\text{Output Value} = (\text{Sensor Output} - \text{Offset}) * \text{Slope} + \text{Bias} \quad (1)$$

Where:

Offset - represents the minimum output of the sensor
Slope - represents the rate of change of the sensor reading
Bias - represents the relation between the sensor and the real world, i.e. zeroing

It should be noted that Equation (1) can have useful physical meaning without the Bias term. The following table summarizes the use of this calibration for the cases of a pressure transducer used for measuring a water level upstream from a broad crested weir, and for a potentiometer used to measure a gate position for a rectangular sluice gate covering a circular pipe, with overlap of the pipe in the fully closed/sealed position.

Table 2. Summary of Linear Calibration

Formula Component	Pressure Transducer	Potentiometer
Offset	Output of the sensor when removed from the water	Output of the sensor when the gate is completely closed and sealed
Slope	Change in the output of the sensor for a given change in the water level above the sensor	Change in the output of the sensor for a given change in the gate position
Bias	Position of the sensor relative to the sill of the weir	The overlap - Difference between the fully closed position and where the orifice is exposed.
Equation 1 without Bias term	Actual depth of the pressure transducer	Absolute gate position with respect to fully closed position
Output Value	Water level above weir sill	Orifice opening.

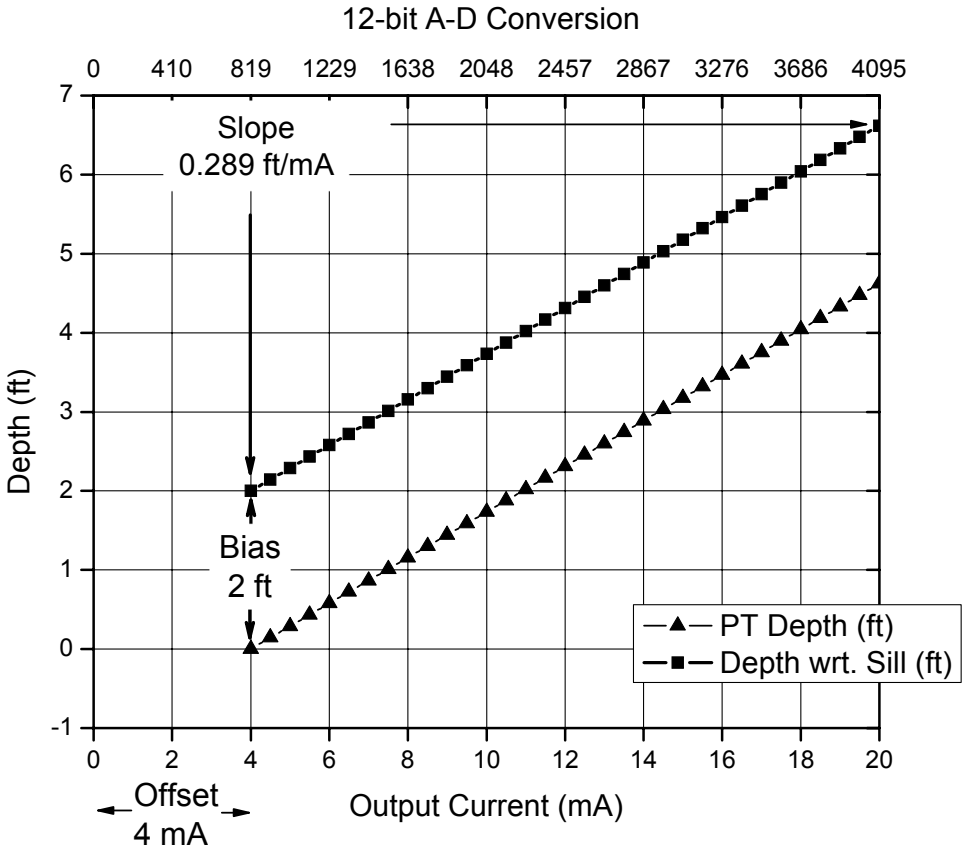

Figure 1. Example of Pressure Transducer Calibration

Figure 1 shows this calibration for a 2 psi (4.62 ft. full-scale) pressure transducer, with an output of 4-20 mA, mounted 2 ft above the invert of a canal pool. Note that the upper axis represents the sensor output from a 12-bit A-D converter.

This calibration method lends itself well to field calibration (Replogle, 1997). As shown for a pressure transducer in Figure 2, the slope can be determined by first taking a reading at the normal operating position and then taking a second reading after displacing the transducer from its operating position by some known distance. By taking measurements at additional offsets from the operating level and calculating a simple regression, the slope may be more accurate. To determine the offset, simply take the pressure transducer out of the water and take a reading. The slope and offset can be calibrated in the office or equipment crib prior to installation at a field site.

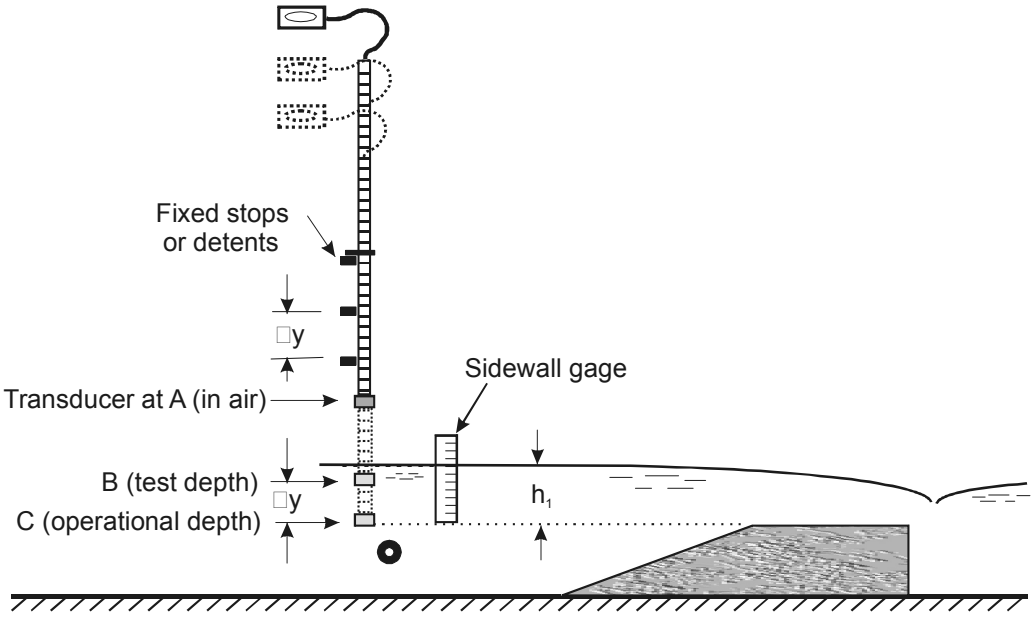

Figure 2. Pressure Transducer Calibration

Finally, to determine the bias, first determine the actual water level by comparing it to a reference point, such as the top of an overflow weir. Place the transducer in its normal operating position, take a reading, and use the slope and offset to calculate the depth of the transducer. The bias is equal to the difference between the actual depth and the transducer depth.

Note that the slope and the offset are properties of the measurement device. Nominal values for these parameters are provided by manufacturers.

Once established, these values can be entered into a SCADA configuration database for use in SCADA software and hardware configuration. It can be advantageous to keep a historic record of these calibration values in order to track seasonal variability in sensor output, to better track sensors that are having problems, and for recovery from of a SCADA computer or field hardware failure.

Operator Displays and Scan Rates

While the underlying data is the foundation of a SCADA system, it means nothing unless the operator is able to discern usable information from the user interface. To optimize computer resources, many SCADA systems rely on multiple timing loops, as shown in Figure 3, to get the information from the field to the user.

In systems designed only for remote supervisory control, i.e. no automatic controls are in place, a normal time-based rate for scanning field sites might be on the order of fifteen or twenty minutes. However, it is common to give the operator the ability to "force" a scan of a particular field unit. Consideration

should be given to timing of the calculations and interface updates to insure that the operator receives timely updates when values change. In the configuration shown in Figure 3, the operator will see the change no more than 90 seconds after it has been read from the field. This timing becomes more critical when implementing centralized control from the SCADA software and automatic forced scans may be needed to provide timely values to control routines.

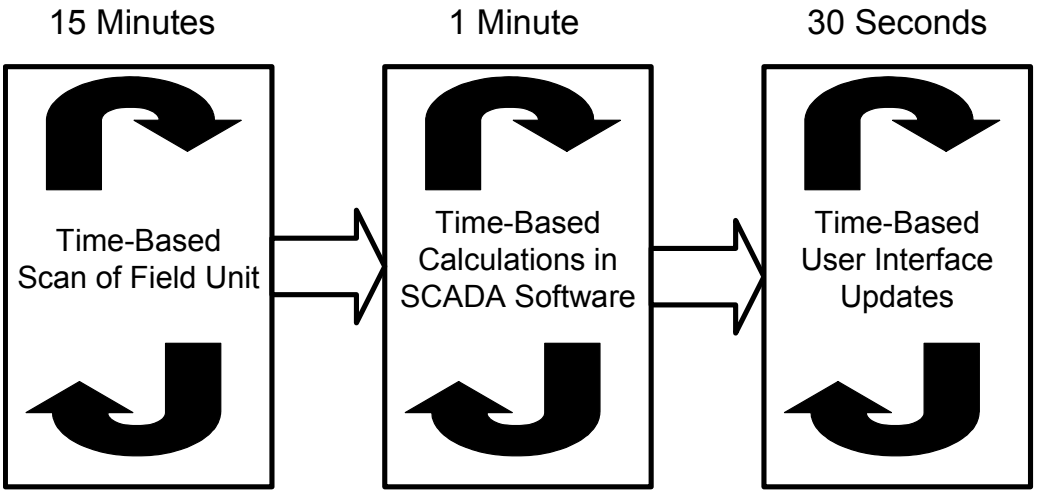

Figure 3. Time-Based Information Path

Communications Infrastructure

Many communication protocols are available for use in a SCADA environment. While formats vary, generally all specify some type of address or ID for a specific device. Additionally, some communication hardware networks, such as spread-spectrum radio networks, have unique addresses for each radio in the network. It is worthwhile to incorporate this information into a SCADA configuration database as accidentally inserting a duplicate address into either layer of the communications environment can lead to problems.

Data Flow and Equations

Some field hardware is capable of returning absolute values that have direct meaning to a SCADA operator. For example, many flow meters are capable of returning a floating-point value of the measured flow. In other cases, a meaningful value has to be calculated either in the field hardware or in the SCADA software. Figure 4 shows the data flow for the calculation of an upstream depth based on the calibration in Equation 1.

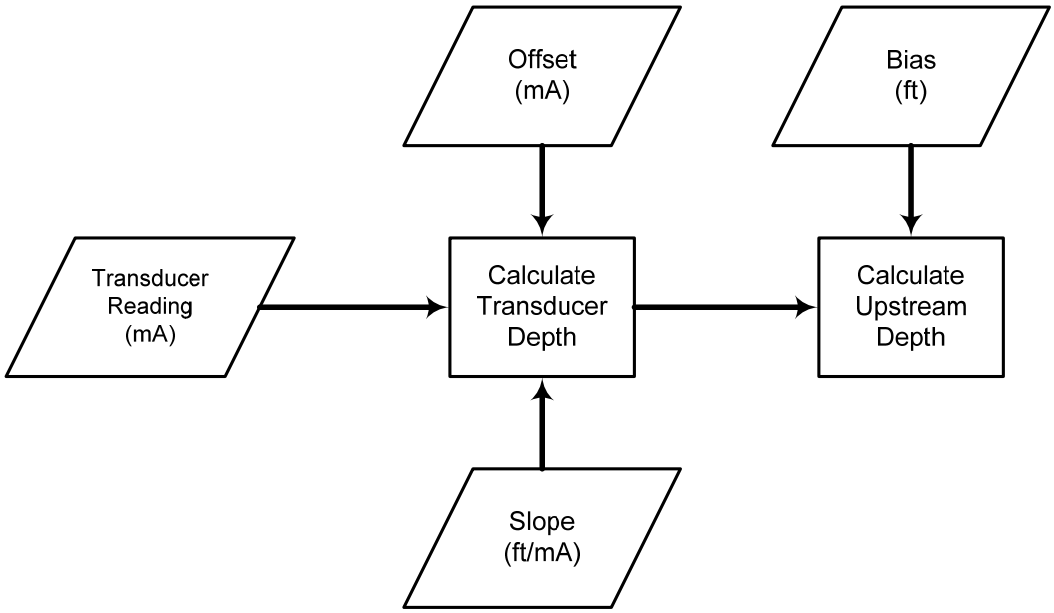

Figure 4. Data Flow for Upstream Depth Calculation

Note that if the transducer reading is returned as a binary integer, the offset and slope are calculated in terms of binary values.

SCADA software packages provide a number of methods for driving these calculations. One option is to execute a calculation on a time basis. Using the timing shown in Figure 3, the communication driver acquires data from a field site every 15 minutes and stores it in the SCADA system. The transducer reading is then checked every minute, which forces calculation of the transducer depth and then the calculation of the upstream depth. The user interface then checks these computed values every 30 seconds and displays the value.

Another option is to execute the calculation on an "exception" basis; meaning that the software executes the calculation only when the input changes. Again, using Figure 3, the communication driver polls a site every 15 minutes. Instead of simply storing the acquired value, it also checks to see if the value has changed. If the value is unchanged, no calculations are executed. If the value has changed, then the new value passed and the calculation chain is executed.

Once the fundamental measurements are calculated, they can be used in other calculations. Figure 5 shows the data flow for the calculation of the combined flow for a control structure consisting of a submerged gate and an overflow weir. Again, these calculations can be carried out in the field or in the central SCADA software. Care should be taken in selecting appropriate equations for computing elements such as weir and gate flows.

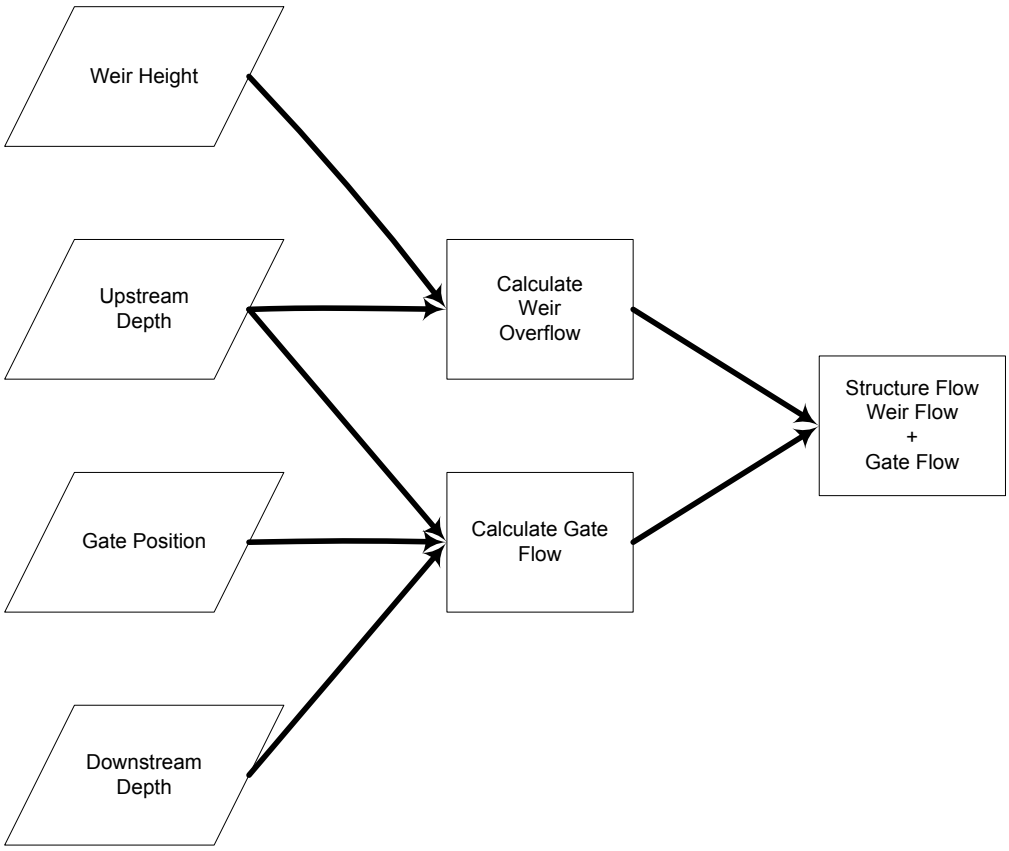

Figure 5. Data Flow Diagram for Calculating Control Structure Flow

Filtering

Filtering should be used with some caution. While it can be helpful to filter the output of a particularly noisy sensor, an improperly configured filter can induce large attenuation or delays in the signal seen by an operator.

In some SCADA software, filtering can be executed in the same manner as other calculations.

Avoid driving a filter calculation on an exception basis. Even for simple moving-average filters, consecutive identical values should be used to compute the filtered value. As a simple example, consider a 4-element data stream of [1,2,2,3]. A three-point moving average would return a value of 2.33. An exception-based three-point moving average would return an erroneous value of 2.

For time-based filtering, make sure that the filtering rate matches the rate at which data is being acquired. If the filter is sampling faster than the data is acquired, then inaccurate duplicates are included in the filter result. Again, using the timing

shown in Figure 3, the filtering should be done on a 15-minute interval. If done more frequently, it will use duplicate values and will be incorrect.

Alarm Limits

An important component of any SCADA system is the capability to alert an operator to problems at field sites through some type of alarming system. A primary type of alarm is a limit alarm for semi-continuous values such as flow rate or water level. The ability to prioritize alarms is helpful. For instance, a flow rate deviation of +/- 10% from an established flow setpoint may warrant a lower priority alarm than a flow rate that exceeds the design capacity of the canal reach.

Rate-of-Change (ROC) alarms are important as well. These alarms are based on a time-rate of change in a particular value. Generally these alarms are set up to examine a particular calculation result or measurement at regular intervals. For instance, a ROC alarm could be configured to examine a water level at 20 minute intervals and signal an operator if the level changed more than 0.3 feet in that time period.

The timing and the allowable incremental change need to be appropriately scaled. For example, to protect canal linings, a district may have a policy of not allowing pool levels in large canals to change more than one foot in a 24-hour period. It wouldn't be appropriate to check for a one-foot change in the level one time each day. The time frame is too coarse. Splitting it up into a 0.014 ft limit in a 20 minute period probably won't work either. The incremental limit is too fine and the alarm will be signaled continuously. Two different ROC alarms may be advisable. One alarm would monitor long term changes and another would catch quick changes due to a stuck gate, a weed plug, or vandalism.

While it isn't necessarily inadvisable to base ROC alarms on filtered values, care should be taken to ensure that the filter doesn't attenuate the real signal to the point that the ROC alarm is overly delayed.

ADDITIONAL CONSIDERATIONS

Data Security

While some data elements such as the width of a gate or the flow capacity of a canal reach probably won't change over time, there are other items that SCADA operators will periodically need to modify. For instance, it is convenient for a central operator to be able to execute a field calibration of a pressure transducer with the assistance of someone at the field site. Using the calibration scheme previously described, the operator would need the ability to modify the offset, slope, and bias of the calibration.

As usual, when granting access to configuration data, there is potential for exploitation. For example, consider a situation where an operator is having trouble keeping a pool level within the alarm limits, is getting tired of hearing the alarm horn, and doesn't have access to modify the alarm limits. One way to silence the alarm is to modify the calibration bias and bring the computed water level within the limits. While this might alleviate the frustration of hearing an alarm horn, the computed water level is no longer accurate. This can cause problems for the subsequent shifts who probably won't know about the calibration change.

Many SCADA software packages incorporate internal security and user management systems into the software architecture. Additionally, some Windows-based SCADA packages are able to run as a Windows Service. This allows the SCADA software to continue to run while a new operator logs in to the system. In either case, the SCADA software can then tailor data access and track data changes and user actions based on the internal user information or the Windows user information.

Historical Data

In order to analyze system operation and for possible liability issues, various data elements in the SCADA system may need to be preserved in a time-based historical database. These might include water levels, flow rates, gate positions, level and flow setpoints, control actions, and the raw data from field devices that would be used to compute other values. Many SCADA software packages allow developers to specify which data elements are stored, and the frequency at which they are recorded. Some time-based historical database formats minimize the amount of storage space needed by only creating entries in the database when values change. When querying the database for historic data, a complete timeline is created from the recorded changes.

Many SCADA packages encrypt historic databases to prevent tampering. This may be an important consideration if there is a chance that an irrigation district may need to use this information in liability litigation.

REFERENCES

Replogle, J. A. 1997. Practical technologies for irrigation flow control and measurement. Journal of Irrigation and Drainage Systems. Kluwer Academic Publishers, the Netherlands 11(3):241-259