2013 ISA WWAC Symposium

Reinventing the Role of the SCADA Historian

Distributed Redundancy, Centralized Access

Blair Sooley, M.B.A, P. Eng.,[1] and Chris Little[1]

[1]Trihedral Engineering Limited, 1160 Bedford Highway, Bedford, Nova Scotia, Canada, B4A 1C1

KEYWORDS

Utilities, Water, Wastewater, SCADA, Historian, Data Logging, Maintenance/Operations

ABSTRACT

As the restless keeper of important SCADA information, the Historian is often overloaded, never appreciated, but sorely missed when unavailable. Indeed, today's culture of increased monitoring, real-time data analysis and frenzied reporting further taxes the traditional Historian Server with access rates hundreds of times that of most business database applications. While improved RAID storage offers some protection from the inevitable hard drive crash, access speeds suffer greatly and a single server still creates a single point of failure.

The solution lies in leveraging server clustering technology with a decidedly SCADA bent, allowing logged data to be distributed across a geographically dispersed network of dedicated SCADA client and server computers that would otherwise be lightly loaded. Moreover, by distributing each piece of data to two or more computers, different levels of redundancy can be applied to different priority data. The role of the Historian evolves into the centralized interface to the data warehouse, directing report queries and new data to the various distributed data storage locations while backup Historians stand at the ready to assume responsibility. This new architecture offers unlimited storage, greater reliability and faster access than ever before.

This paper explores specific examples of how utilities are applying distributed, multi-plant Historian methodologies to meet demands for improved efficiency and regulatory analysis in an era of reduced operating budgets.

ISSUES WITH CURRENT HISTORIAN TECHNOLOGY

The traditional Historian has simply been an instrument for sampling and recording data. Historical data provides information for reporting, situational traceability for upset analysis and historical comparison for process improvement. While increased database storage has offered the opportunity to advance each of these initiatives, most Historian tools have evolved little in their basic application.

Indeed, many SCADA products still use two separate databases, a real-time database for working data access and a historical database for long-term data storage. In these architectures, the Historian's role is to periodically evaluate the real-time database and log data to the long-term database based on a predefined set of rules.

This paradigm suffers from several limitations. First, the Historian evaluates the real-time database regularly whether or not its data has changed. This periodic loading of the computer's CPU is significant and does not scale well, leaning the SCADA industry toward ever more powerful, dedicated Historian server computers. As the number of data retrieval tasks increase, this translates to even greater CPU use and the need for more powerful, dedicated servers.

Secondly, many of these Historians utilize databases such as Microsoft SQL and Oracle; databases designed for complex data storage and analysis with a large number of concurrent user access connections as is common in a business environment. The SCADA environment, however, requires high speed access, large storage of simply formatted data (often just date/time and value), few concurrent connections and the ability to provide time-based datasets.

Thirdly, the Historian typically logs to a single database. This single database is then queried for data from various 3rd party sources and even the SCADA application itself. In the event the database is unavailable, data transfer to and from these systems is interrupted. Data storage is frequently neither redundant nor synchronized.

CURRENT HISTORIAN DATA PROTECTION SCHEMES ARE INADEQUATE

This lack of historical data redundancy represents the most significant concern with the typical Historian. With the collision of wide area network (WAN) client access and advancements in hacking technology, the SCADA Historian is both more accessible and more difficult to protect. In a joint presentation held 2006, the US Department of Homeland Security and the Department of Energy (DHS, DOE, 2006) identified three sources of SCADA cyber threats, Strategic Information Warfare (e.g. terrorism), Direct Cyber Attacks (e.g. disgruntled employees) and General Cyber Attacks (for notoriety/fame.) To address such threats the Global Information Assurance Certification group's (www.GIAC.org) paper *SCADA Networks* (Martin, Bahkto, date unknown) provides an excellent summary of methods currently available to protect SCADA Historians and data network components. In addition to malicious attacks, equipment malfunctions and natural disasters are no less a threat than they historically have been.

Since the hard drive is the primary storage means for historical data, drive failure risks are often mitigated through the use of a redundant array of inexpensive disks (RAID.) Such disks may use data mirroring on two disks (RAID 1,) (Wikipedia topic RAID, data unknown) or striping data and parity information (RAID 5) across several disks to ensure system continuity in the event of single drive failure, or combine both technologies (RAID 10) to allow for system continuity in the event of multiple device failure. RAID drives, however, use a single controller for all read/write actions, which presents a single point of failure. An additional concern is that SATA RAID controllers use randomly oriented disks and suffer performance degradation that increases significantly as the number of redundant drives increases. For example, a read request theoretically takes an average of 33% longer on a two disk array and 67% longer on a 5 disk array over a standard single disk of the same rotation speed (Wikipedia topic Standard RAID Levels, data unknown.) In practice, this degradation proves much higher. This problem is eliminated through the use of synchronized disks, as provided by SCSI RAID controllers.

An off-line backup or archive of the historical database ensures a data snapshot will survive a catastrophic data storage failure and provides for easy offsite data storage. Some databases even allow backup processes to occur without Incurring downtime. The primary concern with such methods, however, is that the backup is only current at the moment it is created. Additionally, restoring a backup takes time and the database may not support restoring at the same time real-time data is being logged. Finally, there is the issue of ongoing media maintenance as new technologies (currently USB drives) eclipse old (3 1/2, 5 1/4, tape drives, ZIP files, CD.)

One imperfect scenario that has been often applied is the use of redundant, independent Historians. Each Historian acts in isolation, sampling and logging data from the real-time database, ensuring that data will continue to be available in the event of a single Historian failure. However, with no synchronization process between the databases, different data may be recorded for the same variable if the Historians are not sampling the same real-time variable at the same moment. Further, if either Historian experiences an outage, the 'available' Historian will continue to log data. Without the two databases being aware of one another, the Historian that experienced the outage will not be backfilled with missing data when it is returned to service. As a result, the same report run for the same period against each Historian database may return different results. This, of course, is unacceptable.

ROLE EVOLUTION - FASTER, SMARTER, FAULT TOLERANT

With an increase in responsibilities and sheer magnitude of stored data, the role of the Historian is being forced to evolve. It must become faster, more scalable, and more fault tolerant. Such changes in role have become a role in itself, and Historians must now *guide* the process of historical data storage and retrieval rather than undertaking the processes themselves.

First, there is the problem of overloading the Historian server's CPU. Consider the typical Historian as a trumpeter and the historical data as a digital recording. The trumpeter is asked to record a song and purchases the sheet music. As he comes to the first note in the first measure, he plays the lowest note on the instrument and compares it to the sheet music. If it is not a match, he proceeds to the next higher note on the instrument, repeating the process until a match is found. Once found, the note is recorded. This process is then repeated for the second note on the sheet music, and so on.

While this scenario may seem silly, it illustrates the downside of sampling data at a regular frequency. It would be more efficient to treat each note occurrence as a specific event. Event-driven execution would suggest that the trumpeter's brain has registered that he is capable of and poised to play a C note. As a C note 'event' (similar to a change in a particular variable value) occurs, he plays the C note and is immediately ready for the next C note event, or any of the other anticipated events, for that matter. This greatly reduces the trumpeter's number of actions and the corresponding load on his brain (the CPU), thus improving response time. Additionally, the reduced brain load allows the trumpeter to more easily handle more complex arrangements of notes (larger historical data recording demands.)

Now consider the case of two trumpeters requested to play the same song, each being recorded separately. Each trumpeter purchases sheet music for the song. However, without dictating a specific arrangement, the trumpeters may have purchased and therefore be playing slightly different arrangements of the same song. While a person listening to the recordings will recognize them as the same song, they may vary in loudness, tonality, chord progressions, even key. Similar differences may occur with historical data recordings, where unsynchronized Historians record data at different times and are unaware of what each other has recorded. This may be acceptable in musical recordings, however, few will accept that their historical data reports may differ depending which data storage is queried.

In essence, if the recordings are to be exactly the same, they must be synchronized. One of the two trumpeters must be designated as a leader (primary historical data server), ensuring both trumpeters play the same music at exactly the same time, and in exactly the same way. The leader selects the arrangement for all to use. If the leader is unavailable, however, the second trumpeter (redundant historical data server) is available to take over all of the leader's functions.

This concept becomes even more important in a system with three or more levels of redundant historical data recordings, where three or more differing versions of the 'same' data can cause great confusion and distrust from the end users. In a musical setting, we'll re-title our leader as the 'First Chair' of an orchestral section and appoint a backup leader, the 'Second Chair', and so on. The Chair hierarchy is common in an orchestra and ensures all musicians in a section know who is currently in charge.

Similar to today's musical recordings, our historical data storage is greatly aided by digital technology. For example, if either musician in a section is unavailable to record a portion of the arrangement, another musician's recording could be digitally copied and inserted (backfilled) into the hole of the incomplete recording with no discernible difference.

GEORGETOWN, KY - SYNCHRONIZATION AND BACKFILL WITH GEOGRAPHICAL SEPARATION

The City of Georgetown, KY, developed its own SCADA application to service the city's two water treatment plants. WTP1 includes a historical database, which is used for primary storage of local data and backup storage of data from WTP2. Conversely, WTP2 includes a 2nd historical database, which is used for primary storage of local data and backup storage of data from WTP1.

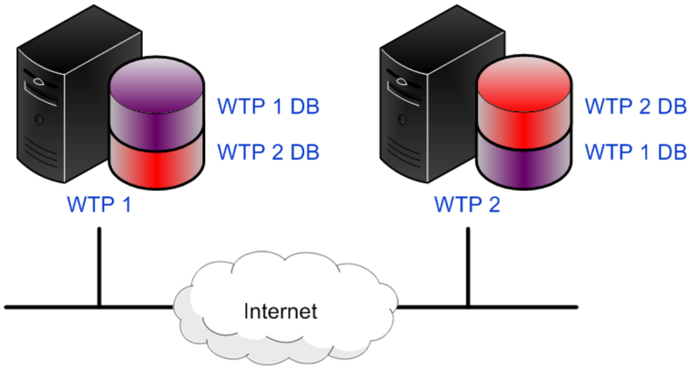

Figure 1: Single SCADA application. Two plants each provide primary storage for local data and backup storage for the others' data.

Both databases are synchronized in real-time across an Internet connection using a secure VPN IPsec tunnel. If either one of the plants' databases is unavailable, the historian still continue to store all data in the remaining database. When the 'outage' database is returned to service, a comparison is done on the database to determine if any data is missing and this data is backfilled.

This architecture is fault tolerant, in that during a network outage, each server will assume the Historian role and log data to the local database. Following the outage, the data backfill will be bi-directional, ensuring both databases are identical. Further, network faults during the backfill process may interrupt the process momentarily but will not result in any loss of backfill data.

CONDUCTING THE ORCHESTRA

The role of the new Historian, then, is to manage synchronization, to guarantee data is backfilled to all replicated storage notes, and to protect the integrity of the data storage. But that's not all.

A large SCADA system's historical data storage can be very complex: Some data is given higher priority, some data is maintained longer than others, and data may be gathered from a variety of sources. The data storage begins to more closely resemble the recording of a full orchestra, with various dissimilar sections - strings, brass, woodwind, percussion - all dedicated to separate parts of the musical arrangement and each with synchronized redundant instruments.

To support this complexity, the Historian becomes a conductor, defining first chairs (primary servers) and second chairs (backup servers) and so on for each section. The Chairs take on the responsibility of synchronizing specific subsections of a musical arrangement (datasets) within their respective sections. Since the conductor handles the high level organization, the individual sections need not concern themselves with one another.

Many hands make light work and each musician is loaded with their specific tasks. However, if we note that traditional SCADA architecture has a maximum of two powerful Historian Server computers doing all the work, we soon recognize that our orchestra would only have two musicians playing all of the instruments. But a typical SCADA system architecture also often includes other lightly loaded computers in the form of I/O servers and client workstations. These workstations are the additional musicians, with excess CPU cycles to collect and synchronize data, and largely unused disk drives for additional historical data storage space.

The result is a cluster of workstation CPUs, each dedicated to management of a portion of the historical data recording, and a cluster of data storage drives that together resemble the RAID configuration discussed earlier, but with unlimited scalability and multi-level fault tolerance. For example, a single application serving two water plants might have historical data storage from a Plant 1 mirrored on a pair of computers (RAID 1) and data from a Plant 2 mirrored on a second pair of computers (also RAID 1.) The entire historical data warehouse, then, is essentially striped across multiple computers (RAID 5.) Combining these methods gives the equivalent of a RAID 10 drive set, while the speed degradation concerns inherent with standard RAID's single-controller architecture have been negated through the use of numerous networked computers' CPUs to handle their own historical data recording.

OCALA, FL - DISTRIBUTED HISTORIAN IN PRACTICE

The City of Ocala, Florida illustrates how a distributed Historian allowed the utility to minimize the number of SCADA servers in their network while improving historical data storage and system redundancy. The application includes six functionally separate SCADA systems, a spray field, three water reclamation facilities, a water treatment plant and a large lift station system, combined into a single, unified SCADA application. All of the systems are connected via a self-healing fiber ring network. Information delivered to each functional areas' operations group is filtered to only include that of concern to the respective area. A unified SCADA approach allows shared security and centralized configuration across the utility.

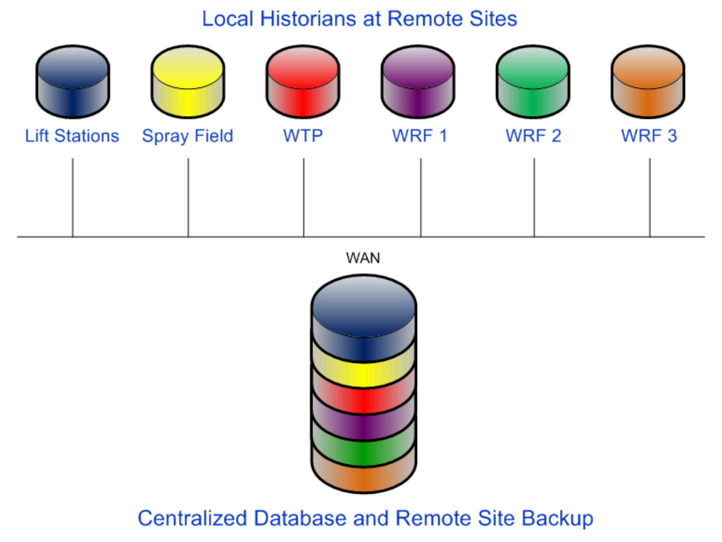

Figure 2: Single SCADA application with distributed historical storage filtered by function and centralized backup.

Each functional area has its own local historical storage which only logs data from the local process. A second, collocated historical backup is unnecessary. Instead, the lift station SCADA server, geographically separated in a different part of the utility complex, is oversized to provide a backup storage location for all six historical databases.

The Historian now has two locations from which to extract any single piece of data. While the backup storage location offers a single access point for all data requests, in the event the backup storage is unavailable, the Historian can still service all data requests by dividing the requests into subparts and requesting the subparts from each of the geographically separated databases. From the consumer's point of view, this activity is transparent. The Historian simply services the request.

The result of this design is a more robust infrastructure with significant savings from a reduced number of server computers, centralized system configuration and management, and lower training costs. Further, consumers are now able to develop reports that combine data from six functionally independent systems.

HELP, THE CONDUCTOR IS SICK! - ALAS, THE SHOW MUST GO ON!

And what if our conductor is unavailable, as our Historian now represents its own new single point of failure for the entire orchestra. This situation is easily remedied through the appointment of a backup conductor (Historian.) Where a single backup conductor is not deemed sufficient, the relative light loading of the conductor role now allows any of the SCADA network's computers to assume the Historian's role, allowing for a 2nd, 3rd or even 4th backup Historian. Essentially, the number of redundant Historians is only limited by the number of available computer nodes in the SCADA system.

STILL WANT CENTRALIZED DATA STORAGE? OKAY, BUT MAKE IT BETTER

In some situations, the classic concept of a primary storage location for all data is still appealing. In the event of catastrophe, this omniscient device can be quickly collected and carted away to safety. But if catastrophe is of concern, it greatly assists the situation to ensure that an online, redundant copy of the data exists in a separate geographical location, such as the plant from where the data originated. In a two plant situation as seen with the Georgetown example above, each plant may keep a primary copy of its own data and a backup copy of the other plants' data. Where three or more plants exist, each may hold a primary copy of its own data and a redundant copy of one or more of the other plants' data. The Historian, our conductor, knows where all of the primary and backup data exist, and can direct the retrieval of this data such that data consumers need have no concern about the details. To the consumer, all data is delivered quickly and from a single source. Complexity made simple.

Taking this a step further, the concept can be leveraged for data buffering at the source, greatly mitigating communications network outage risks. Consider the situation of an application with small, geographically separated water management assets connected via a wide area network. The cost of centralized server-level SCADA at each asset is not viable; rather the utility opts for a centralized SCADA system. For local process visualization at the assets, small HMI's are often used. These panels offer minimal, if any, historical data storage and no synchronization of this data with the central SCADA. Now consider upgrading each of these HMI panels to a small SCADA node that can function as part of the SCADA network when it is available, and autonomously when the SCADA network is unavailable. The cost difference is negligible and reduced training tilts the scale in favor of the small SCADA node. This node is now directed by the Historian to be a historical data storage location for local process data and to synchronize that data with the central data storage. The additional data storage CPU load is easily handled by the small node. Now, in the event of a network interruption between the central SCADA and the local SCADA workstations, the local node realizes that the primary Historian is unavailable and adopts the role itself, continuing to store data uninterrupted. Once the primary SCADA and hence the primary Historian is again available, it relinquishes the Historian role and backfills data from the local node to the central data storage.

This method scales easily, such that any number of local SCADA nodes can leverage unused local CPU and hard drive capability to buffer their own data and act as backup historical data for the central storage.

ONTARIO POWER GENERATION (OPG), ON - LOCAL BUFFERING AT THE SOURCE

Uninterrupted historical data capture in the power generation industry is typically more critical than in the water/wastewater industry, not because the industry is any more critical, but because events happen more quickly and are typically of shorter duration. Indeed, a split second of downtime can miss a cascade of events. As such, OPG's Central Hydro Plant Group uses a report-by-exception protocol to report data

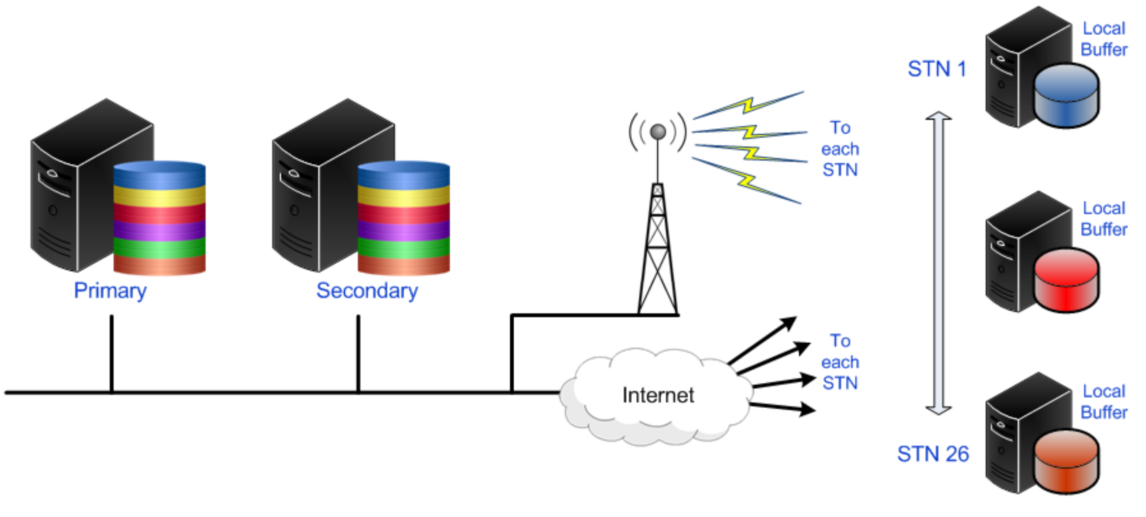

Figure 3: Centralized redundancy with local data buffering at 26 remote stations.

from 26 remote hydro generation stations and record this data to a pair of collocated historical databases located in the central control room. The Historian directs data storage, synchronization services and bi-directional data backfill between the two databases.

While the collocated databases provide redundant storage, a communications interruption of the link between the central servers and any remote station may result in a loss of data reported from the site during this period. To mitigate the likelihood of failure, each link has redundant communications paths; a primary path via Virtual Private Network (VPN) IPsec tunnel and a backup path via IP over cellular. As a 3rd level of redundancy, OPG will be installing small workstations at each remote station, using the workstation hard drives as local historical data buffers. This will provide both backfilling of any missing historical data as well as a geographically separated third level of redundant historical data storage. Further, system offers up to a 10:1 data compression ratio of the synchronization process, allowing backfill from the remote sites without overloading low bandwidth cellular links.

SELECTING A DATABASE

A large business database such as SQL or Oracle is often selected for historical data storage. These powerful beasts are expensive, resource intensive and potentially unnecessary. Many SCADA products have their own low cost or free Native historical database format, tuned to best match their software's capabilities. Indeed, such databases may provide a level of services not provided with business databases, including real-time synchronization services and bi-directional backfilling, important services for SCADA reliability. However, for those who want or have SQL or Oracle, all is not lost. If we review our Historian's role as the conductor, we should note that it doesn't really care what database format is installed on each data storage node, so long as the node understands how to read data from, and write data to, the database. The Historian role, then, transcends the database format, allowing data to be synchronized across several different database formats (SQL, Oracle and Native SCADA) in the same application.

For those who embrace the idea of mixing different databases within the same application, there is an additional benefit to be gained. Each different database format represents its own security hurdle for hackers. As such, an application that stores redundant historical data in more than one format is less susceptible to malicious attacks that cripple the system.

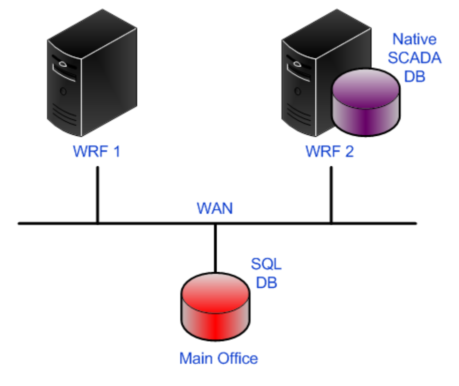

Figure 4: Single SCADA application with mixed historical database formats.

GAINESVILLE, FL - COMBINED DATABASE FORMATS

One such installation is at Gainesville Regional Utilities, FL, where its SCADA Historian logs 6700 data points from a multi-plant SCADA application servicing two Water plants, Kanapaha and Main Street, and 167 lift stations. The application Historian synchronizes data across two historical databases of different formats; the Kanapaha application server logging data to a Native SCADA database format, and the Main Street application server logging data to a SQL Server database located in the main utility offices. Both database have an identical, synchronized copy of the data from all of the infrastructure, ensuring geographically separated protection from catastrophic failure.

This multi-format implementation was selected to meet separate user needs. The first user group, plant operations, typically requests recent historical data (last day, hour) for context regarding recent activities. Response must be fast so as not to burden the application server and slow the operating interfaces. These requests are best serviced by the Native SCADA database installed at Kanapaha, which offers a ten-fold speed improvement over SQL and a five-fold improvement over Oracle (in the SCADA product manufacturer's testing.)

The second user group, the engineering staff, were more interested in long-term analysis, such as identifying seasonal variances and preventive maintenance initiatives. They already had an offsite lightly loaded SQL database installed to service other applications and wanted a single storage location for all of their applications' data. As such, the decision was made to leverage the existing SQL Server database for redundant SCADA historical data storage. Requests from these users to the application Historian would be directed to this database since they typically included large datasets. Since the database was not located on either of the main SCADA operations servers, CPU and memory loading was not a factor.

SERVICING DATA REQUESTS

With all this attention on security and storage paradigms, it's easy to overlook the reason data is recorded in the first place. It is intended to be consumed, and consumers are a varied and typically impatient folk. With data distributed across a number of computers and drives, one would think it a daunting task to gather the bits and pieces for a relatively complicated requirement, such as a monthly summary report. Not so. In this situation, the conductor role provides a single point of connectivity for servicing historical data requests. For instance, a report may require historical data from various parts of various plants. Think of this as a request for a complicated musical arrangement from the audience. Our conductor understands the capabilities of his orchestra and what each section can contribute. The audience (consumer) need not speak with each individual orchestra section to see if each knows the piece (has the data.) Rather, the conductor receives the request and need only review the arrangement to determine if his orchestra has matching capabilities. Indeed, a particular arrangement may require only two or three sections to participate. If strings are required, then he asks the string section to participate. If percussion is not, that section remains idle. Further, the conductor would likely select his First (primary) Trumpet Chair to lead the trumpet section, but is well aware that in the absence of the First Chair, there is a Second Chair, and possibly a Third, each of whom have the capability to lead.

Our conductor also knows what his audience wants to hear, and understands that the notes and songs in his musicians' repertoire may be combined to make wonderful new compositions for consumption by his eager audience. For example, our SCADA consumers may be more interested in the minimum, maximum or average values over a period rather than the raw notes (data), or they may be interested in how many ON/OFF transitions a pump has made since it was last serviced. The historian can make this data available and moreover it can calculate on-demand in order to minimize wasted drive storage space. Essentially, he needn't waste time creating songs (information) his audience has no appetite for.

COST IS KING

And what of the cost? The additional sophistication and benefits of such Historian architecture must surely be greater than ever before. Think again. Throughout the redesign described above, cost is being eliminated rather than added. Expensive, dedicated, server-level data warehouses can be exchanged for workstation-level computers due to the combination of event-driven execution, lighter CPU loading and smaller, distributed data storage across the network. Geographical distribution of redundant data storage mitigates the need for two redundant data storage nodes at each plant. Furthermore, local historical data storage at remote sites offer failsafe data buffering at the source using Native SCADA storage rather than expensive 3rd party databases. And finally, expensive business databases along with their multi-connection licensing schemes are no longer necessary at the central SCADA servers in exchange for these aforementioned Native databases that support data queries using industry standard protocols.

SUMMARY

Regardless the metaphor, whether it be conducting an orchestra or supervising a parts warehouse, the Historian has been promoted to a high level data management position in a growing business. It must be scalable, faster, able to turn data into information on demand, and able to regenerate damaged parts. Its position is an essential part of the SCADA application and must be entirely replaceable without interruption of operational activities or loss of information. To do this, the Historian must be severed from its traditional binds as a single computer-single database manager.

REFERENCES

US Department of Homeland Security (DHS), US Department of Energy (DOE) (2006). *Introduction SCADA Security for Managers and Operators* [Presentation]. Retrieved from http://energy.gov/sites/prod/files/oeprod/DocumentsandMedia/Introduction_to_SCADA_Security_for_Managers_and_Operators.pdf

Martin, T., & Bahkto, A. *SCADA Networks.* Retrieved from https://www.giac.org/cissp-papers/352.pdf

Wikipedia. *RAID*. Retrieved from http://en.wikipedia.org/wiki/RAID, accessed May 03, 2013

Wikipedia. Standard RAID Levels. Retrieved from http://en.wikipedia.org/wiki/Standard_RAID_levels

---

About the Authors:

Blair Sooley is a Pre-Sales Account Manager with Trihedral Engineering Inc. He holds a Bachelor in Electrical Engineering from Dalhousie University in Halifax, and an MBA from St. Mary's University. Blair has been working in the controls industry for 18 years and in the water and wastewater sector for 10 years.

Christopher Little is a native of Halifax, Nova Scotia and a graduate of Dalhousie University. Over the last seven years he has written numerous articles and has been published in Pumps and Systems, Environmental Science & Engineering, and Arab Water World. His work also appears regularly on www.wateronline.com.