

## **ENTELEC 2004 HOW TO AVOID GETTING STUCK WITH A DEAD-END SCADA OR LEAK DETECTION SYSTEM**

By Donald B. Ashton  
UTSI International Corporation

### **SYSTEM NEEDS CHANGE**

SCADA and Leak Detection system requirements are continually changing. Sometimes change is expected, such as accommodating gradual modifications to the pipeline infrastructure, and sometimes the change is unexpected, such as in a corporate merger, asset sale/purchase, or new government regulations affecting pipeline operations or security. Sometimes change is dictated by the inability to obtain compatible replacement or enhanced (faster) hardware, or by the inability to obtain hardware maintenance at a reasonable cost. Change could also be dictated by the inability to add a desired software feature, or to obtain reasonably priced software support.

However, regardless of the source, SCADA and Leak Detection systems need to be able to accommodate change. No one wants to depend on a system that cannot be changed or supported for reasonable amounts of money.

### **PROJECT TEAMS ALWAYS ADDRESS CHANGE**

The need for change is not a new concept for SCADA and Leak Detection project teams. Bidders on new projects are usually restricted to established vendors who are expected to provide quality support and a commitment to keep their systems current. Virtually every bid specification contains words similar to the following:

"The vendor shall provide regular software updates and system enhancements. Vendor support personnel shall be available to add custom features after system acceptance, at a reasonable cost."

Hardly ever has a bidder been known to take exception to such requirements in a bid specification. Often, but not always, SCADA and Leak Detection customers have naively put their faith in these promised updates and continuing support as the sole means of meeting their future needs to support change.

## **UPDATES AND CONTINUING SUPPORT HAVE NOT BEEN THE NORM**

Contrary to the hopes and expectations of the buyers of SCADA and Leak Detection systems, and to the assurances given by vendors in proposals, most vendors of these systems have left their customers with Dead-End systems far earlier than the users had thought possible. Most of the leading SCADA systems of the early 1990's became Dead-End Systems in less than ten (10) years.

What are the characteristics of a Dead-End System?

- ◆ No periodic usable updates from the software vendor,
- ◆ No vendor capability to add custom features at a reasonable price,
- ◆ No reasonably priced upgrade path to a system that does have regular updates,
- ◆ No ability of the pipeline company to modify the source code to meet custom requirements, and
- ◆ Often, no reasonably priced way to replace the system with a new system due to customized, unusual features of the system, or overwhelming screen development costs.

## **PROMISES OF UPDATES AND SUPPORT ARE NOT ENOUGH!**

Many pipeline companies accepted the promises of regular software updates and continued product support. They have blindly believed these empty promises were sufficient protection against getting stuck with a SCADA or Leak Detection system that is impossible to modify and prohibitively expensive to replace. History has shown that such assurances are woefully inadequate.

In fact, there are at least three (3) ways that a pipeline company can avoid getting stuck with a Dead-End SCADA or Leak Detection system:

1. Be lucky enough to choose a product whose vendor really provides regular updates and continued support,
2. Obtain a source code license with right-to-modify privileges and develop software support expertise, or
3. Structure the system and the in-house support so that system replacement is as manageable as possible.

The first method has been discussed. The remainder of the paper discusses the other two (2) methods of keeping from getting stuck with a Dead-End system.

## **GET THAT SOURCE CODE!**

There are many different positions regarding whether pipeline companies should obtain source code rights with their SCADA and Leak Detection systems.

### Some Arguments Against Getting Source Code

- ◆ Many vendors strongly discourage customers from purchasing source code. The vendors give various explanations of their position, including the danger that competitors will learn their "secrets," a position that vendor support and warranties are incompatible with customer access to source code, and sometimes just a dictatorial edict that "We don't ever provide source code."
- ◆ Some pipeline companies have had bad experiences with source code. A scenario that has happened more than once is that a user makes significant edits to the source code and then belatedly discovers that attractive enhancements offered by the vendor are no longer available because his system has diverged too far from the "standard product."
- ◆ Source code can be a very high cost item.

### Some Arguments In Favor of Getting Source Code

- ◆ You will be in good company. Many of the leading pipeline companies in the United States have obtained source code rights with their SCADA and Leak Detection systems, and utilize the source code to enhance the security and efficiency of their operations. In some cases, these companies have extended the useful life of their systems far beyond what would have been possible without source code rights.
- ◆ Source code ownership provides the user with the ability to perform quality control on new system releases prior to putting the new releases into service on the on-line system. Most of the aspects of new release testing can be performed without the benefit of source code, but minute details often arise where the source code is invaluable.
- ◆ The user is able to determine the detailed internal logic used to accomplish complicated functions. This capability is sometimes useful when a configuration parameter does not seem to have the desired effect. Source code is especially beneficial to users who write their own applications and utilize the SCADA or Leak Detection system's API. API calls are seldom fully

- documented, partly because the vendor's developers have access to source code to determine precise functionality.
- ◆ The source code becomes most valuable in the unfortunate, but all too frequent, situations where the software vendor is unable or unwilling to provide the desired level of support and enhancement for the SCADA or Leak Detection system. Source code ownership can provide the user with the ability to add significant functionality, such as new communication protocols. Without source code, a need for a new protocol might lead to untimely retirement of the entire SCADA system.
  - ◆ Corporate IT organizations have taken over the SCADA support duties at many pipeline companies, and IT organizations ordinarily do not insist on source code rights when purchasing new software. However, SCADA and Leak Detection software differ in some important ways from most other IT-supported software. The degree of reliability and uptime required on SCADA and Leak Detection systems is often an order of magnitude greater than for other IT software packages. The userbases of SCADA software packages are also much smaller than those of most corporate software products. Therefore, the requirement for SCADA and Leak Detection source code rights is very defensible.

Some companies determine that the purchase of right-to-modify source code rights with a new SCADA or Leak Detection system is absolutely necessary to insure that future system support will be available. It is not really feasible to take the position that source code is only necessary when a vendor becomes financially unstable, or when support for a software product seems to be in jeopardy. Since source code rights are often a high-dollar item, the only opportunity to obtain corporate approval for the purchase may be in conjunction with the original system order. Additionally, proposal time is the time when the vendor has the most motivation to sell source code rights at a reasonable price. Source code rights have little-to-no marginal cost for the vendor, so the price charged is very much a function of how much the vendor wants to make the system sale.

## **AVOID SOURCE CODE ESCROW PITFALLS**

SCADA and Leak Detection customers, who desired assurance that they would not be left without access to source code in the event that the vendor becomes unwilling or unable to support the system, have sometimes required that source code be placed in an escrow account. Source code escrow at a third-party location appears in many ways to provide a valuable service. In theory, the escrow agreement provides security against vendor failure, at a reasonable cost. The vendor also reduces the danger of source code proliferation.

In practice, escrow agreements have not provided the expected degree of protection for the customer. There are two (2) areas where source code escrow agreements have often failed:

1. Vendors sometimes refuse to release the source code to the customer, even after active "support" for the software product has ended. The vendor may claim that support is still available and thus refuse to release the source code.
2. A more common problem is that the source code in the escrow account becomes obsolete and almost unusable over time. Unless the source code in escrow is updated every time changes are made to the on-line system, then the escrowed source code is not able to replicate the system that is in operation. An additional concern is that, unless make-files, compilers, linkers, and debuggers are escrowed with the source code, it may be impossible to modify and rebuild the system with the source code once it is taken out of escrow.

If a customer chooses the source code escrow route, he should be very careful about the wording in the contract. The contract should spell out explicitly what conditions will result in the release of the source code. The contract should also specify that every time a change is made to the executables on the on-line system, that those executables be generated from the source code and the utility programs that are stored in the escrow account. The reader will appreciate that a properly maintained escrow account requires a fair amount of effort and diligence on the part of the customer and the vendor as well. Since source code escrow does not provide many of the benefits of source code purchase, the escrow approach may not be the best solution for all customers.

## **SIMPLIFY YOUR NEXT SYSTEM UPGRADE!**

One very sensible way to avoid being stuck with a Dead-End system is to keep open your option to replace your SCADA or Leak Detection system at a reasonable price. Some might scoff at this idea, since the main goal when a system is purchased is to make it great, rather than to be concerned about replacing the new system with another one. However, consider the following factors:

- ◆ The useful life of many SCADA and Leak Detection systems is ten (10) years or less.
- ◆ Mergers and acquisitions often result in consolidation of control centers. And, this often involves replacing recently installed systems.
- ◆ The same techniques that keep a pipeline company well poised to replace its system also keep the company well poised to take maximum advantage of its current system.

Some of the methods of controlling the expense and complexity of future system upgrades and/or replacements are as follows:

- ◆ Set and maintain rigorous standards for database point names. Many procedures to simplify database cutovers and screen development are greatly simplified when the database point names are standardized. When similar pipeline stations have identical point names, except for the characters representing the station name, the process of producing the new screens, configuration files, etc., is greatly simplified. Standardized database point names also promote better understanding by operators and field personnel.
- ◆ Utilize screen templates for display screens. Then continue to update and enhance the templates throughout the life of the SCADA or Leak Detection system. Screen templates are often used when a new pipeline is built, but it is common to abandon the template approach soon after pipeline startup. When templates are abandoned, the time required to re-implement the SCADA screens on a new SCADA system may be several times the time that was required to initially implement the screens. Rigorous adherence to the use of templates can minimize the time required to re-implement screens on a new system. Template use will also minimize the chance of errors creeping in when new stations are added to an existing SCADA system.

The concept of putting in a lot of diligence now to avoid high costs in the future is very much in line with pipeline operations mentality. For example, the benefit of diligent cathodic protection procedures is to prevent pipes from failing many decades in the future.

- ◆ Maintain detailed and up-to-date documentation of the features of your system, and the missing features that you would like to have. Most users, and even most vendors, do not have a complete list of the features of their systems. Such a list will be a great aid in selecting a new system.
- ◆ Seriously consider keeping the communication handling software totally separate from your SCADA or Leak Detection system, and using an industry standard interface between the communication front end and the system host. One of the ways that some pipeline companies get very deeply committed to obsolete SCADA and Leak Detection systems is by tightly coupling unique and complex protocol handling software into their host computer systems.

One drawback to this approach is that it becomes more difficult to provide the pipeline operator with meaningful diagnostic indications of failed communication links. This issue can be addressed by providing good diagnostic capabilities on the front-end communication processors, or by passing diagnostic indications from the front-end processors to the SCADA system for inclusion in the SCADA system's screens and logs.

- ◆ Avoid the temptation to combine the GUIs of separate systems (like SCADA and Leak Detection) into any kind of a "Common MMI," "Universal Back End," or any such futuristic sounding concept. This approach was a highly touted goal several years ago. However, a few implementations of the concept highlighted serious drawbacks. One of the biggest challenges is to be able to upgrade all of the applications and operating systems simultaneously in order to provide continuous reliable performance.

A more workable approach is to integrate GUIs only to the extent necessary for effective operation. Selected variables can be passed from the database of one system to the other system's database, and then the imported variables can be displayed or logged using the native functions of the receiving system. Once an issue is brought to the attention of the pipeline operator, the native GUI of the ancillary system will be accessed by the operator.

- ◆ When you are buying a new SCADA or Leak Detection system, use industry standards to the greatest extent possible. BUT - research the standards to be used closely to be sure that they are not dying away. Purchasing a system that adheres to such standards as X.25, LAT, OLE, or SDLC will make your future life much harder, not easier.

## **WHAT IF YOU ARE ALREADY STUCK WITH A DEAD-END SYSTEM?**

Some of the steps to take when purchasing a new SCADA or Leak Detection system have been discussed. What can a pipeline company do if it is already stuck with a SCADA or Leak Detection system that has no prospect of future updates and is almost prohibitively expensive to replace? Is there nothing to do except bury your head in the sand?

If a pipeline company already knows that software updates will never be coming, but it does not absolutely need to start replacing its system, some valuable steps can be taken to prolong the life of the system and minimize the expense and complexity of system replacement when it does become necessary:

- ◆ Investigate whether source code rights can be purchased at a reasonable price.
- ◆ If there is no document describing the features of your system and the missing features that you would like to have, make it now.
- ◆ Consider isolating any custom parts of your system, such as a unique communication processor, so that it does not need to be replaced simultaneously with the rest of the system.

- ◆ Get a grip on your screen displays. Try to divide the displays into "families" and to determine the feasibility of using templates to re-implement the screens when the time comes.
  
- ◆ Keep your options open on what new system to purchase for as long as possible. The offerings and the relative health of the vendors can change rapidly.