

Successful Automation's Best Kept Secret

Roy Kok, Vice President of Marketing and Sales, Kepware Technologies

Sure, the engineer is proud of his HMI and the plant manager pleased with his KPI dashboard and how effectively the plant is running with the implementation of his new MES solution. The historian and analytic solution get all the credit when a plant control crisis is averted. Does anyone ever think to thank the lowly driver? Without data, the HMI is just a pretty picture, and the best analytics in the world are only ideas without any basis in fact. Plus, you spend tens or hundreds of thousands on that, while the driver budget isn't even a line item on the spec sheet. The communication arteries and veins reach out to all areas of your plant, sending and receiving data, efficiently and tirelessly ensuring your ability to acquire, analyze, and adapt to the situations at hand.

History Lesson

In the days of DOS, drivers were part of the primary application. For HMIs, the device driver was provided by the HMI vendor and would only work with that HMI. Windows came with technology to share information between applications through a technology called DDE (Dynamic Data Exchange), however, the prior investment in custom drivers was too great to throw away. In 1995, the OPC Foundation was created, with the idea to leverage the latest Microsoft technologies, to develop an interoperability standard enabling automation applications to exchange data. Finally, there would be a high performance standard that drivers could leverage for connectivity to more than one vendor.

Vendors quickly adopted the OPC standards; their client applications became OPC-enabled so they could leverage third party drivers and products. Independent driver developers could now find a wider market for their drivers, generally developed as part of some system integration effort. Thus, an industry was born.

While a step in the right direction, this was not the model of perfection. While a standard exists for interoperability, it is up to the developers to adhere to that standard and test. Drivers developed by different developers will have different behavior, user interfaces, methods of operation, diagnostics, etc.

A driver developed for one specific application will not necessarily excel in all other applications and unless there is a significant volume for the application of a driver, the long-term cost of maintenance can become prohibitive, resulting in a driver marketplace delivering varying levels of performance, reliability, functionality, and overall quality. Due to the maintenance costs involved, many vendors have chosen to migrate to an outsourced driver model, leveraging the high volume and industry-wide experience of a dedicated driver development company.

In February of 2008, the OPC Foundation introduced a new level of certification— an OPC Foundation authorized independent test laboratory located in Germany, offering exhaustive testing over a period of several days to prove all aspects of OPC

Successful Automation's Best Kept Secret

Published on Industrial Maintenance & Plant Operation (<http://www.impomag.com>)

conformance, resulting in a “Certified OPC Compliant” logo. This logo shows that the product, and the company behind it, has reached a level of quality that is to be expected, indicating that this driver is likely to be supported now and into the future.

Managing Disruption

In addition to getting data from point A to point B, drivers need to be designed for performance, ease of use, reliability, and optimum operation in the event of a disruption in operation; the latter being a major point, often overlooked in the development of communications.

Devices generally offer a variety of mechanisms for the acquisition of information—supporting single variable reads, reads of blocks of data, or the ability to subscribe to data and receive unsolicited updates:

- The best drivers will navigate these options for the user; auto selecting the method based on data requirements.
- Performance needs to account for the priorities of various tasks.
- Writing information to devices needs to be done efficiently— guaranteed in periods of stress.
- High reading demands cannot override write commands, yet writes cannot dominate.
- The demands of a communication driver should not overly impact the operation of the PC on which it is running.
- Applications often have tag counts into the hundreds of thousands, some updating frequently, others only when diagnostics are active. Drivers must be optimized for multi-threaded operation, must allocate memory effectively, and must clean up after themselves to avoid impacting other processes running in the computer.

Reliability, Ease Of Use

Ease of use includes configuration menus being simple and self-explanatory plus help menus that are detailed and context sensitive. The driver should deliver features for auto-configuration wherever possible. Many devices today contain the details of their configuration either within the device itself, or in a programming file that a self-configuring driver can readily decode, often triggered by a configuration Wizard and automatic. A likely scenario is to configure one portion of a process, perhaps one of many production lines, then use copy and paste, or import and export tools, to replicate the configuration with any necessary tweaks. Reconciling the configurations of different drivers from different vendors, another challenge, can be resolved by selecting drivers from a vendor that has focused on consistency across their driver library in operation and configuration tools.

Since our automation systems are the core to the world's infrastructure, reliability is a must and can be achieved by experience, the reuse of proven code, testing with industry solutions, interoperability meetings, and internal practices to develop and properly QA a driver solution before it is installed on site. Many drivers are born out

Successful Automation's Best Kept Secret

Published on Industrial Maintenance & Plant Operation (<http://www.impomag.com>)

of a system integration need, then repurposed as a standard offering which can lead to low cost upfront, but explosive costs down the line. It is very easy to make a driver work and perform well in the best of conditions, but there are many types of communication failure modes that are often overlooked by the casual driver developer.

Drivers communicating through wireless modes may receive garbled messages due to static, storms, etc. Drivers may be communicating with hundreds of devices, some working, some not. It takes attention to detail in the design of communication buffers, timeout designs and polling strategies to maximize operations under adverse conditions, such as delivering tuning parameters for auto-demotion features, enabling a driver to demote the polling of a failed device.

The best drivers have been designed to handle all of the above, and deliver this functionality consistently across the broadest suite of protocols, giving process engineers one source for all of their device connectivity. Next time you are considering a driver, think about how well your application would run if it failed, and allocate your budget appropriately.

Source URL (retrieved on 10/26/2014 - 4:02am):

http://www.impomag.com/articles/2009/01/successful-automation%E2%80%99s-best-kept-secret?qt-recent_content=0