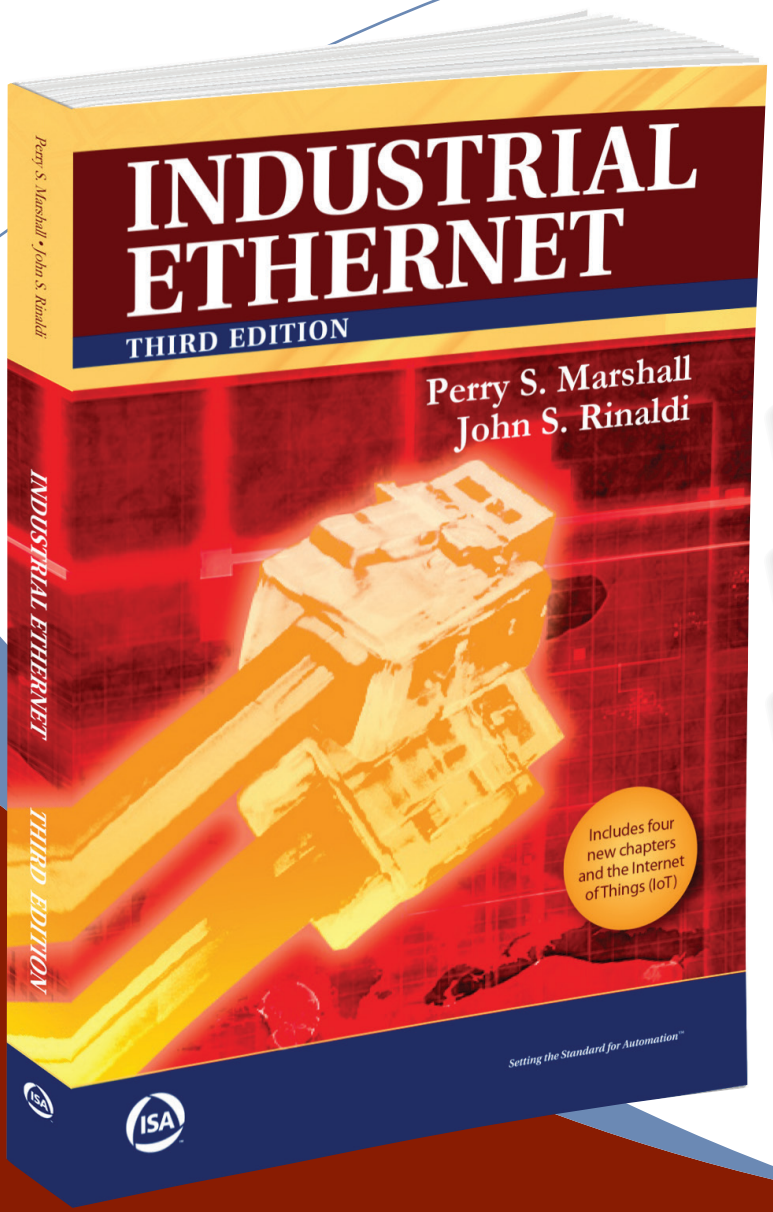
Standards
Certification
Education & Training
Publishing
Conferences & Exhibits



eBook available!

Table of Contents

View Chapter

Buy the Book

Setting the Standard for Automation™

Industrial Ethernet

How to Plan, Install, and Maintain TCP/IP Ethernet Networks: *The Basic Reference Guide for Automation and Process Control Engineers*

Third Edition

By Perry S. Marshall and John S. Rinaldi

Contents

14.0

Factory Floor/ Enterprise Communications

If you have paid any attention to factory automation over the last few years, you have noticed the ever-increasing emphasis on connecting the factory floor to the enterprise. There are many good reasons for this. Some of the reasons are internal: efficiency, productivity, higher quality, and the like. Others are driven by external requirements. Large customers, such as the Walmarts of the world, are demanding higher levels of integration with their suppliers. Regulators are increasing their demands for manufacturers to report on their production processes. Corporate attorneys are "suggesting" that manufacturers archive more data about their production processes.

It wasn't always like this. In the old days (10 years ago?), the production department was a completely separate entity from the rest of the corporation. There was little-to-no electronic data transfer between the production machines and the company's business systems. Production was a black box. Labor and raw materials went in one end, and finished product came out the other end. Most of the communication was carried out

using paper: paper production reports, paper inventory levels, paper raw material usage, paper quality reports, etc. People keyed this information into business systems that used sales data to order raw materials and adjust production levels for the next production cycle—again using paper systems.

Today, the aim is for instantaneous closed-loop communication. As units of product are consumed in the field, that information gets reported back to the machine that made it. The production machine checks its raw material inventory levels and on-hand finished product, and then schedules more production. It automatically transmits orders for any raw materials it needs to supplier machines. All automatic. All without human intervention.

That is the plan anyway. In practice, it is pretty hard to get there. We do not have the luxury of ripping out all the production machines and replacing them with new, fully integrated machines with high-speed communication mechanisms. Instead, we have to do piecemeal implementations: upgrading and replacing systems one by one as time and funds allow. It is a marathon, not a sprint, to the goal of fully automated systems.

There are many factors impeding progress on our path to fully integrated production systems. Security, of course, is key—the more integrated and connected your production process is, the more vulnerable you are to mischief and worse. Another factor is the difficulty of replacing fully capitalized and functional systems that are well understood and perfectly operational but lack the integration required for tomorrow's manufacturing vision. Yet another is the mutual lack of understanding of IT integration by today's control engineers and manufacturing by today's IT people.

14.1 Tight versus Loosely-Coupled Systems

The distinction that many people from manufacturing and IT miss is that there is a key distinction between the systems on the factory floor and the enterprise system. This is the difference between what is called *loosely-coupled* systems and *tightly-coupled* systems. These are not new concepts, but I don't think they have been examined in the light of the current trend towards the integration of factory floor and enterprise systems.

Factory floor systems can be labeled tightly-coupled. Systems that use PROFIBUS, PROFINET IO, DeviceNet, EtherNet/IP, or any Modbus version have a strict architecture. These are really just I/O producers and consumers, despite what some folks at the trade associations might want you to believe. Let's look at the main characteristics of tightly-coupled systems:

- **A Strictly Defined Communication Model** – The communication between these systems is inflexible, tightly regulated, and as deterministic as the communication platforms allow.
- **A Strictly Defined Data Model** – The data model (really an I/O model for most of these systems) is predefined, limited, and inflexible.
- **Strictly Defined Data Types** – The data types transported by these systems are limited, predefined, and supported by both sides. There is no ability to send data in an open and universal format.

We could look at any of the factory floor protocols, but let's take EtherNet/IP as an example. EtherNet/IP has a strictly defined communication model. A scanner uses a precise communications model in communicating with its adapters. The adapters are preconfigured: all data exchanged is predefined,

and nothing changes without human intervention. The data exchanged is part of the adapter's predefined object model, and the data is formatted in a way supported by both the scanner and the adapter.

Tightly-coupled systems provide much needed, well-defined functionality in a highly specific domain. Expanding operation to other domains or trying to provide more general operation is difficult. Making more generic data and functionality available requires significant programming resources that results in a very inflexible interface.

That is why tightly-coupled systems are wrong for enterprise communications. Can they be made to work for a specific application? Yes. But to get there requires a tremendous effort and results in a difficult-to-maintain, inflexible system that is extremely fragile. These systems are difficult to maintain as any small modification anywhere along the line can cause a failure.

Loosely-coupled systems, on the other hand, provide exactly the right kind of interface for enterprise communications. Loosely-coupled systems decouple the platform from the data, decouple the data from the data model, and provide a much more dynamic mechanism for moving data. Loosely-coupled systems have these characteristics:

- **A Widely Used, Standards-Based Transport Layer** – Messages are transported in loosely-coupled systems with open, widely-implemented, highly flexible transports layers: TCP and HTTP.
- **An Open, Platform-Independent Data Encoding** – Data is encoded using an open standard data encoding like XML that can be processed by any computer platform.

- **A Highly Extensible Operating Interface** – The interface between loosely-coupled systems is flexible and extensible. SOAP (the Simple Object Access Protocol, which is rapidly being replaced by REST) is the main interface, and it provides a highly flexible mechanism for messaging between loosely-coupled systems.

Essentially, what I've described here is web services. The web services architecture is the backbone of everything we do on the Internet. It is extensible, flexible, and platform independent—all required for the ever-expanding Internet.

The challenge is to how to best migrate the tightly-coupled factory floor architectures to the loosely-coupled web services architecture of the Internet. It is difficult to migrate today's technologies for transferring I/O data like Modbus TCP, EtherNet/IP and PROFINET IO. It can be done, but it often results in brittle systems that require too much support and cost too much time and money. Integrating these technologies with loosely-coupled enterprise technologies takes massive amounts of human and computing resources. In the process, we lose lots of important metadata, we lose resolution, and we create security concerns. These factory floor systems were not designed to be highly secure. Using the factory floor protocols for enterprise data collections creates systems that are a house of cards.

Because of the discontinuity between the factory floor systems and the enterprise systems, opportunities to mine the factory floor for quality data, interrogate and build databases of maintenance data, feed dashboard reporting systems, gather historical data, and feed enterprise analytic systems are lost. Opportunities to improve maintenance procedures, reduce

downtime, and compare performance at various plants, lines, and cells across the enterprise are all lost.

14.2 OPC UA for Factory-Enterprise Communications

The solution is OPC Unified Architecture (UA), a new architecture for moving information between manufacturing and the enterprise. OPC UA can live in factory floor world and the enterprise world.

OPC UA is about reliability, security, and most of all, easily modeling *objects* and making those objects available around the plant floor, to enterprise applications, and throughout the corporation. The idea behind OPC UA is infinitely broader than anything most of us have ever thought about before.

It all starts with an object. An object that could be as simple as a single piece of data or as sophisticated as a process, a system, or an entire plant. It might also be a combination of data values, metadata, and relationships. For example, let's consider a dual loop controller: the dual loop controller object would relate variables for the set points and actual values for each loop. Those variables would reference other variables that contain metadata like the temperature units, high and low set points, and text descriptions. The object might also make subscriptions available to get notifications on changes to the data or the metadata for that data value. A client accessing that one object can get as little data as it wants (single data value) or an extremely rich set of information that describes that controller and its operation in great detail.

OPC UA is, like its factory floor cousins, composed of a client and a server. The client device requests information. The server device provides it. But, the way that the OPC UA server pro-

cesses the information is much more sophisticated than the process performed by an EtherNet/IP, Modbus TCP, or PROFINET IO server. An OPC UA server models data, information, processes, and systems as objects and presents those objects to clients in ways that are useful to vastly different types of client applications. And better yet, the OPC UA server provides sophisticated services that the client can use, like the Discovery Service, a service that locates available OPC UA devices.

14.3 Ten Things to Know about OPC UA

OPC UA is the future and the perfect technology to bridge the chasm between loosely- and tightly-coupled systems. Here are 10 things you need to understand about OPC UA:

1. **OPC UA is not a protocol.**

   It is a common misconception that OPC UA is just another protocol. That could not be further from the truth.

   A computer protocol is a set of rules that govern the transfer of data from one computer to another. Even though OPC UA also specifies the rules for communication between computers, its vision is more than just moving some arbitrary data from one computer to another. OPC UA is about complete interoperability.

   OPC UA is an architecture that systematizes how to model data, model systems, model machines, and model entire plants. You can model anything in OPC UA. OPC UA is a systems architecture that promotes interoperability between all types of systems in various kinds of applications.

2. **OPC UA is the successor to OPC (now referred to as *OPC Classic*).**

OPC UA solves the deficiencies and limitations of OPC Classic, a technology built on the now obsolete Microsoft COM. In today's world, we need to move data between all sorts of embedded devices, some with specialized real-time operating systems and software, and enterprise/Internet systems. OPC Classic was never designed for that.

OPC UA is the first communication technology built specifically to live in that "no man's land" where data must traverse firewalls, specialized platforms, and security barriers to arrive at a place where that data can be turned into information.

3. **OPC UA supports the client-server architecture.**

   We are all familiar with technologies that have a superior/subordinate relationship, often with one master to many slaves. That is not true of OPC UA clients and servers. In OPC UA, a slave can be configured to accept connections with one, two, or any number of clients. A client device can connect and access the data in any number of servers. It is much more of a peer relationship in OPC UA, though, like other technologies, servers simply respond to requests from clients and never initiate communications. In practice, many devices are being built to easily support peer relationships implementing both client and server functionality.

   Another unusual and interesting aspect of the client/server relationship is that in OPC UA, a server device can allow a client to dynamically discover what level of interoperability it supports, what services it offers, what transports are available, what security levels are supported, and even the type definitions for data types and object types. These characteristics make an OPC

UA server much more sophisticated than the servers for many of the technologies you have worked with in the past.

4. **OPC UA is a platform-independent and extremely scalable technology.**

   Unlike OPC Classic, OPC UA is designed from the ground up to be platform independent. The only requirements for OPC UA are Ethernet and a mechanism to know the current date/time. OPC UA is being deployed to everything from small chips with less than 64K of code space to large workstations with gigabytes of RAM.

   All the components of OPC UA are designed to be scalable, including security, transports, the information model, and its communication model. Several security models are available that support the level of security appropriate for the device's resources and processor bandwidth.

5. **OPC UA integrates well with IT systems.**

   OPC UA servers can support the transports used in many traditional IT-type applications. Servers can connect with these IT applications using SOAP or HTTP (the foundation of the data communication used by the World Wide Web).

   OPC UA servers can also support XML encoding, the encoding scheme used by many IT-type applications. It is likely that most servers in the factory floor automation space will not support XML encoding due to the large amount of resources required to decode and encode XML. However, many servers in that space will

support OPC UA Secure Conversation, a more efficient binary encoding that uses fewer resources.

6. **OPC UA provides a sophisticated address space model.**

   The address space model for OPC UA is more sophisticated than EtherNet/IP, PROFINET IO, Modbus, or any of the industrial or building automation protocols. The fundamental component of an OPC UA address space is an element called a *node*. A node is described by its attributes (a set of characteristics) and interconnected to other nodes by its references or relationships with other nodes.

7. **OPC UA provides a true information model.**

   The ability of an object node to have references to other object nodes that further reference other object nodes to an unlimited degree, provides the capability to form hierarchical relationships that represent systems, processes, and information—an *information model*.

   An information model is nothing more than a logical representation of a physical process. An information model can represent something as tiny as a screw, a component of a process like a pump, or something as complex and large as an entire filling machine. The information model is simply a well-defined structure of information devoid of any details on how to access process variables, metadata, or anything else contained within it.

   Many trade groups—including many in the oil and gas industry, the building automation industry, and PLC standards organizations, and others—are using the information model capabilities of OPC UA to define

information models for their application domains. They are using OPC UA for the standard transports, security, and access to their data models.

8. **OPC UA extends factory floor communications.**

   Instead of a factory floor protocol, you could say OPC UA is web services for automation systems, that it is SOA for automation systems. SOA is basically the same thing as web services. That is fine if you are an IT guy (or gal) and you understand those terms. You have some context.

   But if you're a plant floor guy, it is likely that even though you use web services (it is the plumbing for the Internet). And it is just as likely you may also say, "Why do we need another protocol? Modbus TCP, EtherNet/IP, and PROFINET IO work just fine." The answer is that OPC UA is not like EtherNet/IP, PROFINET IO, or Modbus TCP. It is a completely new paradigm for plant floor communications. It is like trying to explain EtherNet/IP to a PLC programmer in 1982. With nothing to compare it to, it is impossible to understand.

   OPC UA lives in parallel with these technologies. It doesn't replace them. It extends them by bringing in new functionality, creating new use cases, and driving new applications. In the end, it increases productivity, enhances quality, and lowers costs by providing not only more data, but the right kind of data to the production, maintenance, and IT systems that need it when they need it.

9. **OPC UA is a certifiable standard.**

   Like many other technologies, there is a process to validate that an OPC UA device conforms to the standard. And like many other technologies, there is documentation to certify that devices pass the OPC UA certification test suite. A successful compliance test results in an electronic test certificate being transmitted to the device. Client devices can then access the device and get the electronic certificate documenting its status as a certified OPC UA device.

10. **OPC UA is still a developing technology.**

    There is a technology adoption life cycle, and OPC UA is following that cycle. The first systems were available just a few years ago. Like any other technology adoption, there are the innovators, closely followed by the early adopters. If the technology is successful, the next group, the early majority adopters join in and then the technology reaches its peak adoption.

    OPC UA is the preferred communications protocol of numerous trade associations, of major vendors (such as SAP SE, one of the world's leading IT companies), and is the core for the entire German Industry 4.0 effort (the German government, industry, and educational systems combined effort to develop new manufacturing technology).

14.4 Reference

For a free book on OPC UA written by the author of this book, go to the web page listed below and leave a message asking for the book *OPC UA: The Everyman's Guide to OPC Unified Architecture* by John Rinaldi: www.rtaautomation.com/contact-us/.

14.5 Summary

There is much to gain by integrating the factory floor and the enterprise. There are efficiencies, flexibility, and vast amounts of data that can be mined to improve factory floor processes. For example, the vast majority of energy usage in a factory is from its motors but there is often no ability to access data on which motors consume the most energy, which are efficient, and which are inefficient. Enormous savings are available if this sort of information can be extracted from the factory floor.

But combining factory floor systems with the enterprise is challenging. The cultures, systems, and technologies are very different. Enterprise systems are flexible, open, and loosely coupled while many factory floor systems are closed, proprietary, and tightly coupled. OPC UA is a new technology which may be the bridge between the factory floor and the enterprise. OPC UA offers a unique set of capabilities that may unlock the data and information on the factory floor and yield the kinds of savings described above.