**INTRODUCTION**

In the last 30 years we’ve had any number of new technologies and products introduced into the automation market. We’ve seen the introduction of the programmable controller replacing relay logic, electronic drives replacing mechanical line shafts and high speed networks replacing direct wiring just to name a few. But nothing has rivaled the attention on the Industrial Internet of Things (IIoT), or the Internet of Things (IoT) as it’s known across all sectors.

It’s definitely a strange time for us in industrial automation. I don’t recall seeing articles in The New York Times or The Wall Street Journal on the trends in electronic drives systems. For the last few years, it’s been difficult to not hear about IoT. All the big Silicon Valley players like Microsoft, Oracle, IBM and Amazon are heavily invested in IoT. One of the most significant players in industrial automation, GE, is focusing on using their Predix Cloud-based service platform for the collection of data from big machines like jet engines and locomotives.

It’s hard to believe something so pervasive, with so many tentacles into our lives, can be another technology bubble, but it’s hard to deny that it appears that way. Whenever you have vendors, customers, the media, academia, government and Wall Street promoting a trend as a revolution, you can’t help but think of the Dutch tulip panic of 1637, the technology bust of the late 1990s, and the real estate boom of the early 2000s.

It’s clear that if you’re in the business of providing solutions for the general population or even a farm, mobile apps, electronics for the home, wearables, medical devices and such that there is great promise for the IoT.

But what does this all mean for system integrators, PLC programmers and control engineers, especially those of us who use Allen Bradley programmable controllers to make machines work on the factory floor. How will the IoT affect us?

**WHY THE TALK ABOUT IOT ON THE FACTORY FLOOR?**

If you listen to the smart people at Microsoft, the killer app for the factory floor is predictive maintenance. I’ve never bought that and never will. Predictive maintenance works for processes that provide an opportunity to act on the predictive maintenance information. Jet engines are a perfect example. Engine flight data and improved scheduling of engine maintenance has allowed Delta Airlines to reduce the number of jets held in reserve by nearly two-thirds. That’s a savings of hundreds of millions of dollars.

That’s not possible for a lot of factory floor applications. On the factory floor we have continuous processes to maximize the return on expensive manufacturing systems. With 24 hour, 7 day a week operation and a lot of continuous processes, there is no opportunity to act on predictive maintenance information. Instead, it’s more efficient to wait for the maintenance shutdown and replace everything. Predictive maintenance isn’t the killer app that many of the Silicon Valley companies think it is.

The big benefit of IoT on the factory floor is the ability to connect the factory floor with the enterprise, with MES systems, ERP systems, suppliers and customers. The ease of connecting operational systems (programmable controllers and the like), the ability to build applications where machines can connect with suppliers or collect currently inaccessible data (energy data, for example) are all benefits. If that all sounds like déjà vu, I agree with you. That’s what we’ve always tried to do.

But what’s never been attempted is the automatic configuration of machine components. That means, turn a machine on and the components all talk to one another, identify what data to pass to each other and when to do it. That might sound like science fiction but it is, in fact, one of the overriding goals of Germany’s Industry 4.0 - another way to express IIoT or IoT - machines that can configure themselves.

But maybe we’re getting ahead of ourselves. Let’s look at what IoT really is.

**WHAT THE INTERNET OF THINGS REALLY IS**

At its very core, the Internet of things is electronic devices – devices with microcontrollers – sharing their data with other devices. If you’re an automation professional, you reaction is “what’s new about that? Haven’t we been doing that for a long time?” And you’d be correct in saying that. We’ve had devices sending data to other devices on the factory floor now for 30 years. Anyone ever heard of Modbus?

But IoT is more than that. What’s new is that IoT is being driven by a set of technologies that make it easier than ever to not
just move data but move it incredibly fast, archive it, analyze it, visualize it and turn that data into information. We now have easy access to incredible analytics, incredibly fast networks, protocols built to collect data from thousands of devices, inexpensive Cloud servers, fancy visualization tools and the ability to add Ethernet or some form of wireless communications to low-cost devices at an ever decreasing cost.

Our problem is that we live in an applications paradigm (the factory floor) where we have devices that either weren’t designed to share their data – think of the old chart recorders, photo eyes, and proximity sensors – or could only share data in limited ways. And I’m not just talking about sensors. Drives, robots, programmable controllers and most other devices on the line today aren’t all that much better. Tens of thousands of devices with Profibus DP and DeviceNet communications have limited bandwidth and aren’t designed to share a lot of data. Even devices with EtherNet/IP or Profinet IO aren’t usually able to easily provide data beyond that which resides in its cyclic IO packets.

Our challenge is to take advantage of all the power of IoT on the factory floor, in this very restrictive environment where there just isn’t any ROI for replacing equipment.

How do we do that? Let’s first get a handle on the technologies that comprise IoT. Unfortunately, there’s a lot of them.

**FACTORY FLOOR TECHNOLOGIES FOR IOT**

The really unfortunate aspect of IoT is that there are literally hundreds of protocols used in IoT applications, and everyone has their favorite. A number of them support collection of data from thousands of devices (AMQP, MQTT and DDS). Others are designed for specific media (Zigbee, ISA 100, 6lowpan). Others are simple and well understood (HTTP and HTTPS). Others are not protocols but architectures (REST and OPC UA). There are deficiencies and limitations to each of them. Each of them supports a different level of performance and security.

When reading the list you’ll probably notice that you’ve been doing factory floor IoT for years without even knowing it (Yes, you should add a new skill to your resume)!

**XML**

**WHAT IS IT?**

The eXtensible Markup Language, known as XML, is a data language which communicates by sending files of ASCII characters from one system to another. XML is verbose but also simple and human readable. XML is a meta-markup language. That means that data in an XML document is surrounded by text markup that assigns tags to the data values. Each data value, when taken together with its distinguishing tag name, is an XML Element—the basic defining unit of an XML document. An entire collection of elements forms the XML document.

**WHAT ARE ITS ADVANTAGES AND DISADVANTAGES?**

XML is perfect for IoT applications which are sending data to enterprise applications. Almost all Microsoft and enterprise applications can easily ingest XML data. The deficiency of XML is that it’s verbose and requires significant resources: resources that many embedded devices find difficult to provide.

**HTTP**

**WHAT IS IT?**

The Hypertext Transfer Protocol (HTTP) is the connectionless, stateless protocol that is used every time we access a web page. It is included here as more than a few vendors implement it as a very simple way of moving data between automation devices and IT and IoT applications. HTTP is a request / response protocol. A Client establishes a TCP connection with a Server and sends an HTTP request to the Server. The request generally includes a URL, the protocol version, and a message containing request parameters, Client information, and sometimes a message body. The Server responds with a status line which includes the protocol version, a response code, and a message body. An HTTP message contains either a GET request to retrieve information from the remote system, a PUT (or POST) request to send information, or a HEAD request, which returns everything the GET request does except the message body.

**WHAT ARE ITS ADVANTAGES AND DISADVANTAGES?**

HTTP is a very simple technology and many vendors have built applications on top of it to move data from automation devices to IT applications. It is not difficult for vendors to customize HTTP GET and POST messages and add custom protocol information in the message body. By building applications that use these protocols, these vendors create easy-to-use mechanisms for moving automation data between IT systems and the factory floor.

However, a huge problem is that HTTP closes the connection when the request is complete. A new request means that another connection must be opened leading to more overhead. And what’s really unfortunate is that HTTP, like most other IoT protocols, provides no information model, no services other than the raw GET and PUT, and no standardized mechanism to publish data when new data is available. Despite these limitations, HTTP’s popularity and simplicity make it a very popular protocol for IoT applications where building a proprietary implementation is desirable.
MQTT (AMQP, DDS)

WHAT IS IT?
Message Queuing Telemetry Transport (MQTT) is another mechanism for moving data around the factory floor or from the manufacturing environment to the Cloud, MQTT, as well as Advanced Message Queuing Protocol (AMQP) and Data Distribution Service (DDS) are all designed to meet the challenge of publishing small pieces of data in volume to lots of consumers’ devices constrained by low-bandwidth, high-latency, or unreliable networks. MQTT supports dynamic communication environments where large volumes of data and events need to be made available to tens of thousands of servers and other consumers.

Though frequently publishing small pieces of data in huge volumes doesn’t sound applicable to many factory floor applications, it needs to be considered as the huge IoT system providers (Amazon, Microsoft, Oracle, GE and IBM) all either support MQTT, or protocols similar to MQTT, to ingest data into their IoT hubs.

The heart and soul of MQTT is its publish/subscribe architecture. This architecture allows a message to be published once and go to multiple consumers, with complete message decoupling between the producer of the data/events and the consumer(s) of the messages and events.

WHAT ARE ITS ADVANTAGES AND DISADVANTAGES?
MQTT is a very simple way of distributing information from lots of publishers to lots of consumers. The ability to support thousands of devices distinguishes it from some of the alternatives already discussed. The publish/subscribe technology of MQTT is vastly superior to that of other technologies, as much more bandwidth is consumed by the more complicated publish/subscribe implementations. It is extremely lightweight, reliable, and adapts well to low-resource devices. Broker devices, which some view as a disadvantage, manage the connection between the publishers and consumers.

The disadvantage to MQTT and nearly all of these protocols is that they are simply transport protocols. There is no “schema” functionality which a receiver can use to understand the content of a packet. The receiver has to “know” what every byte it received is.

OPC UA

WHAT IS IT?
The challenge of the factory floor is how to best integrate the tightly-coupled factory floor architectures of today’s programmable controllers and Ethernet networks to the loosely-coupled Web Services architecture of the Enterprise and the Internet. Integrating these technologies with the loosely-coupled enterprise technologies takes massive amounts of human and computing resources to get anything done. In the process, we lose lots of important metadata; we lose resolution and we create fragile and brittle systems that are nightmares to support. And don’t even ask about the security holes they create. These systems were not designed to be highly secure. These systems are a house of cards.

Because of the discontinuity between the factory floor and the enterprise, we lose 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. Opportunities to improve maintenance procedures, reduce downtime, and compare performance at various plants, lines, and cells across the enterprise are all lost.

The solution to this challenge may be OPC UA. OPC UA can live in both the world of the factory floor and the enterprise. OPC UA is about reliably, securely, 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 it is infinitely broader than anything most of us have ever thought about before.

And 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 be a combination of data values, metadata, and relationships. Take a dual loop controller: the dual loop controller object would relate variables for the setpoints and actual values for each loop. Those variables would reference other variables that contain metadata like the temperature units, high and low setpoints, and text descriptions. The object might also make available subscriptions to get notifications on changes to the data values 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 what the UA Server does is much more sophisticated than what an EtherNet/IP, Modbus TCP, or Profinet IO Server does.

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. Better yet, the UA Server provides sophisticated services that the Client can use, like the Discovery Service, used to find OPC UA servers and identify their capabilities.

**WHAT ARE ITS ADVANTAGES AND DISADVANTAGES?**

OPC UA is not a protocol. OPC UA is an architecture for moving data around the factory floor and the enterprise. It has a number of unique features. It is the only architecture that completely separates the encoding, transporting and message security from the messaging layer and the address space. That provides an opportunity for many organizations to implement their particular data model and messaging scheme (BACnet, for example) while using the powerful and easily integrated security, transports and encoding offered by BACnet. OPC UA also provides features like scalability, device discovery, publish-subscribe and a modeling system vastly more powerful than any in use today.

As with all technologies, OPC UA has its disadvantages. It is complex. It can be difficult to implement, and there are versions with new functionality that are not going to be backwards-compatible.

**REST**

**WHAT IS IT?**

Unlike the other concepts described in this paper, REpresentation-al State Transfer (REST) is not a protocol and not a technology, but actually an architectural concept for moving data around the Internet. The REST architecture, or a RESTful interface, is simply a very flexible design, usually built on top of HTTP, for Client devices to make requests of Server devices using well-defined and simple processes.

In REST, the concept of how devices on a network function is different than the conceptual view of a network for most other networking technologies. We usually think of a network as a set of devices that provide some specific set of services. A Modbus device, for example, provides a specific set of services like Read Coil, Read Holding Register...etc. In most technologies used in Industrial Automation, there is some set of predefined services that Client devices must learn, implement, and use to access the resources of a device. That sort of architecture works well in our limited paradigm automation systems but it doesn’t work well in the world of data transfer to the Enterprise and the Cloud.

REST is resource-centric instead of function-centric. In the RESTful architecture, a Server is viewed as a set of resources, nouns if you will, that can be operated on by a simple set of verbs like GET, POST, UPDATE, and the like. This architecture yields a much more flexible mechanism for retrieving resources than the limited function-centric kinds of technologies we’ve used in the past.

**WHAT ARE ITS ADVANTAGES AND DISADVANTAGES?**

REST is a very good alternative for building simple IoT applications. It is simple to understand, easy to implement, but less functional than some of the other alternatives. As a simple mechanism to move factory floor data to an IT application or Cloud Server, REST can be a good choice. You can implement a factory floor Server that provides a REST interface, and define Java objects, XML, or CSV as the delivery format for your data. It won’t be real time—but you don’t always need real time data.

**AB PROGRAMMABLE CONTROLLERS & THE IOT**

If you’re an integrator, distributor, control engineer or other automation professional, your customers are demanding more integration with the Enterprise. You’ve always integrated automation devices with Windows and Linux applications, but now you need to transfer that factory floor data to enterprise-based applications and Cloud-based applications where that data can be archived, visualized, processed and analyzed. Some of your customers even want forward integration with their customers and backwards integration with their suppliers.

That’s a big challenge. Sometimes the data you need is locked up in a device and not easily accessible. Sometimes it doesn’t really exist. Other times it’s available on some old, proprietary and currently unsupported network like DH+. But often it’s in a programmable controller.

If you’ve got a new controller from Siemens, Beckhoff or Wago, it’s likely that it supports communication using OPC UA. With native support for OPC UA in Windows 10, you can get the data you need pretty easily and seamlessly. But that’s pretty unlikely as those controllers are currently a small part of the market.

How do you create some sort of IoT application if you have an old Allen-Bradley controller? What if you have a ControlLogix? Even though ControlLogix, let alone PLC5, SLCs and MicroLogix have no inherent ability to move data to the Enterprise, there are a few possibilities.

**THROW ANOTHER COMPUTER AT IT**

This is the way we’ve always done it. Use an OPC Classic driver, RsLinx or the RTA Tag Client to move data from the Allen-Bradley programmable controller into a Windows environment. From there, you write your own application to move those data table entries to a local database, a database on another server or some application on an enterprise or Cloud server.

That is an IoT application but it’s not pretty. Often we have to program the PLC to collect some data not inherently part of its control loop, like energy data. That data is mapped to the PLC data table, then gets transferred to the Windows environment and then gets
transferred someplace else. You can lose resolution, the original data ID and format can get lost, there’s no timestamping and you have no metadata describing the data. It works but it’s guaranteed that the process will break and you’ll be in there troubleshooting and fixing it.

**ADD A MODULE TO THE RACK**

Another method is to buy an “IoT module.” There are vendors that are selling in-rack solutions for ControlLogix. Softing has one called the eATM tManager. It is a very powerful solution for ControlLogix-based applications. It’s highly integrated with the PLC’s data table and it can move vast amounts of data very quickly to Oracle or SQL databases. It’s a pricey solution, but it’s your only one if you have huge amounts of data.

**USE AN “EDGE” GATEWAY MODULE**

Edge is used here in quotes as it’s another term with no real definition. There a vendors now starting to offer these “edge” gateways that can move automation and building data using IoT protocols.

Some are simply gateways that use EtherNet/IP, Profinet IO and other factory floor protocols to gather data and then send it using an IoT protocol like MQTT or OPC UA. That type of edge gateways work fine if the data you need is available over the Ethernet network, but it’s nowhere near a perfect solution.

Some data (motor drive energy data for example) isn’t usually included in a control packet. So first the device has to make the data you want available over the network, and not all devices do. Second, since the gateway would have to explicitly open a connection with the device having the data and then send a command to get the data, you’re going to be using bandwidth dedicated to the machine operation. Do a significant amount of that for a lot of devices and all of a sudden, the operation of the machine degrades. Some manufacturers are installing information networks alongside the operational networks just for this scenario.

Another problem is that a lot of these gateways can use the Ethernet network protocols but don’t have the capability to access the data tables of the PLCs. Much of the data to send to an IoT application is locked up in that PLC and you have to find a gateway that knows how to go and get it from that data table.

**Real Time Automation** has several products in development that are going to meet this need. One is able to take the data table entries from an Allen-Bradley programmable controller and push them to enterprise and Cloud applications using simple file transfers like XML and CSV. Another can push data on demand using an HTTP Client and JSON. Another can OPC UA-enable your Allen-Bradley programmable controllers to communicate using OPC UA.

Both of these products vastly increase the connectivity of your PLC, SLC, MicroLogix and Logix programmable Controllers and make it much easier to build IoT applications.

**WHAT TO DO NEXT**

There is no perfect solution for every application problem. The platform, quantity of data elements to access, timing and other considerations must all be considered before choosing a solution. Sometimes, it’s an odd communication interface, an off-brand printer, meter or barcode reader. Sometimes, it’s a performance issue. Sometimes, it’s a question of what hardware can support a software application. And sometimes what’s needed is some guidance on what the technology is and where it’s going before implementing a new system.

If you need to increase the connectivity of your AB programmable controllers, no matter how old that PLC might be or what your application is, Real Time Automation can assist you in understanding the complex world of the Internet of Things and networking technology as it applies to AB PLCs.

Call 800-249-1612 to discuss your application or click on http://www.rtautomation.com/contact-us/ to email us.

**ABOUT THE AUTHOR**

John Rinaldi is Chief Strategist, Business Development Manager and CEO of Real Time Automation (RTA) in Pewaukee, WI. John is an Electrical Engineer with a Master’s Degree in Computer Science (MS CS). His work experience includes Procter & Gamble, Kimberly-Clark and Rockwell Automation.

John and his team of expert automation specialists at Real Time Automation supply industrial and building network connectivity software and products all over the world. With a focus on simplicity, support, fast delivery of product, expert consulting and tailoring for specific customer applications, RTA is meeting customer needs in applications worldwide.

John is now not only a recognized expert in industrial networks and an automation strategist but a speaker, blogger, the author of over 30 articles on industrial networking and the author of multiple books.

John’s latest book, released in February 2019, is an in-depth guide to EtherNet/IP. You can download the first three chapters for free by clicking HERE.