EtherNet/ IP in 90 days or less:
A plan for product developers
By John Rinaldi and Jamin Wendorf

Industrial Ethernet is one of the hottest topics in automation today, and for good reason. Ethernet has become the worldwide de facto standard for networking, and its potential to standardize communication on the factory floor is tremendously appealing. Of the available industrial application layers, EtherNet/IP (EIP) offers the most functionality to the widest audience.

EIP is enjoying enthusiastic interest in the industrial automation community. Many automation vendors already have customers who are attracted to the advanced capabilities, increased speed, and perceived lower cost over current sensor bus networks. With General Motors pursuing a tiered approach to networking using EIP, the gauntlet has hit the floor, and automation vendors inside and outside automobiles are being drawn into the web. Frankly, today it is a mistake not to support EIP.

Once you decide that your products must support EIP, the big question becomes “how can I get it done and get it done quickly?”

Do you have excess engineering resources to commit to the project? Can your existing opportunities and customers for EIP be put off until next year? Is your engineering staff intimately familiar with the specification? Have they followed the nuances discussed by the developers in the ODVA System Joint Special Interest Group on EIP?

If you’re having difficulty with any of these issues, then this article was written for you. It details the technical information you need to know and gives you practical steps for adding EIP to your existing product in the next 90 days.

What is EIP, and why do I need it in my product?
EIP is a high-level industrial application layer protocol for industrial automation applications. Built on the standard TCP/IP protocol suite, EIP uses all the traditional Ethernet hardware and software to define an application layer protocol that structures the tasks of configuring, accessing, and controlling industrial automation devices. EIP uses standard Ethernet hardware and software to make truly open systems a reality in the automation business. Your customers are not just asking for connectivity, they absolutely expect and require it.

Rockwell Automation has equipped many of its programmable controllers with EIP connectivity. Since its inception in 2000, more than 1000 industrial automation professionals have attended Ethernet enabler sessions put on by the ODVA, and hundreds more have attended detailed product development training. If you aren’t already in this game, your competitors are already a step ahead of you.

What is causing this enthusiasm for EIP?
First, there is a palatable feeling among customers in many different industries that Ethernet is easily understood, that components and tools are readily available and that it will be less expensive than current automation networks. Whether or not any of this is true is open to discussion and argument. What cannot be argued is that a large number of major automation customers, such as General Motors, want Ethernet. All the major programmable control vendors are planning to support one or more application layers on Ethernet.

EIP is a very important technology, not only because it addresses most of the technical concerns of Industrial Ethernet, but also because it is backed by Rockwell/Allen-Bradley. This setup gives it an automatic advantage in the marketplace.

Therefore, EIP will be successful for many of the same reasons that DeviceNet has been successful. The momentum is strong, and its eventual success is inevitable.

A technical summary of EIP and CIP
The EIP protocol is an application layer protocol based on the Control and Information Protocol (CIP) layer used in DeviceNet and ControlNet. EIP devices are represented on industrial networks using the identical object model structure as DeviceNet and ControlNet devices. The object model structure, the way a device presents itself to the network, is the one of the real keys to the growing popularity of EIP. For the first time, there can be a common infrastructure between the sensor, control, and information networks of an installation. With this common messaging structure, a device can just as easily request data from another EIP device as a device on DeviceNet or ControlNet.

In addition to the object model implementation, EIP also supports unconnected messaging and many of the same I/O messaging sequences as DeviceNet. Change-of-state, cyclic, and a form of polled messaging are found in both EIP and DeviceNet.

EIP: All the advantages of DeviceNet, combined with the Internet
Though similar to DeviceNet, EIP provides vastly improved functionality over DeviceNet. Where DeviceNet is primarily a master-slave technology, EIP is a client-server technology with support for connections to multiple clients and multiple servers.

EIP devices can use any of the protocol suites inherent in the underlying TCP protocol, including mail service (SMTP), browser support (HTTP), and other services.

EIP also provides support for broadcast and multi-cast messaging. Broadcasting can be used by nodes as identity messages so that a client device like a programmable controller can be alerted to their presence on a network. Multi-casting is a way of distributing data to many clients and is a valuable way of coordinating separate devices, cells, or factories.

The core communication protocols within EIP are the encapsulation and CIP protocols. The encapsulation layer encapsulates CIP and other protocols. Knowing that many devices will continue to support standard and custom legacy protocols, the encapsulation layer is designed to carry...
these protocols. By encapsulating custom serial protocols, EIP becomes a valuable tool to transition today’s legacy protocols and less robust serial communication links to faster and more reliable technology (see Figure 1).

**What exactly is the CIP protocol?**

CIP is a communications protocol for transferring automation data between devices. In the CIP protocol, every network device represents itself as a series of objects. Each object is simply a grouping of the related data values in a device. For example, every CIP device is required to make an identity object available to the network. The identity object contains related identity data values called attributes. Attributes for the identity object include the vendor ID, device type, device serial number, and other identity data. CIP does not specify at all how this object data is implemented, only what data values or attributes must be supported and that these attributes must be available to other CIP devices.

The identity object is an example of a required object. There are three types of objects defined by the CIP protocol:

- **Required objects**: These are objects required by the specification to be included in every CIP device. These objects include the identity object, a message router object, and a network object.

- **Application objects**: These are objects that define the data encapsulated by the device. These objects are specific to the device type and function. For example, a motor object on a drive system has attributes describing the frequency, current rating, and motor size. An analog input object on an I/O device has attributes that define the type, resolution, and current value for the analog input. Just as in DeviceNet, these application layer objects are predefined in a device profile for a large number of common device types. A large number of profiles for many device types have been defined. Supporting a device profile allows a user to easily understand and switch between vendors with common device types.

- **Vendor specific objects**: These are objects not found in the profile for a device class and are termed vendor specific. A vendor includes these objects as additional features of the device. CIP provides access to these vendor extension objects in exactly the same method as application and required objects. This data is strictly of the vendors choosing and is organized in whatever method makes sense to the device vendor.

In addition to specifying how device data is represented to the network, the CIP protocol specifies a number of different ways in which that data can be accessed such as cyclic, polled, and change-of-state.

**What other protocols should you consider?**

The advantages of the CIP protocol layer on an EIP network are numerous, but what other Ethernet protocols are worth considering? There are numerous application layer competitors to EIP including Modbus/TCP from Groupe Schneider, Profinet from Siemens, HSE Fieldbus from the Fieldbus Foundation, and IDA from a group of European vendors. Modbus/TCP is the only one of this group with a significant following.

Modbus/TCP is a much simpler protocol than EIP. Its easy implementation is its greatest strength and its greatest weakness is its limited functionality. Also built on top of the TCP Modbus portion is the connectionless Modbus/TCP. Once a TCP connection is established, there is no additional overhead. As in Modbus, each server is represented by a very simple set of registers and coils and responds to the traditional Modbus command set.

The simplicity of Modbus/TCP is a real liability. Without the object model structure, complex network devices such as drives, motion controllers, and other devices lose all of their internal structure in the flat Modbus/TCP network representation. Without the common structure, there is often no commonality between the same devices from different manufacturers and little interchangeability. With support for only command/response messaging, there is no built-in structure to transfer data synchronously or on an as needed basis. (See Table 1.)

**How to implement EIP quickly and effectively**

Once you’ve made a decision to add EIP connectivity, you have a number of ways to proceed. The choices vary in time to market, supportability, resource requirements, and price.

**Use an off-the-shelf serial gateway**

An off-the-shelf serial gateway is the least economical approach with the fastest time to market. To implement a serial gateway, your product must support a common serial communication protocol such as Modbus, DF/1, or even dumb ASCII. Data from your product will transfer to the gateway at serial baud rates of 1200-19,200.

Other than time to market, typically a week or two, there are few real advantages to this approach and many negatives. Gateways require an additional footprint, they can be costly, they rarely support messaging characteristics needed for the data, and the gateway vendor name

---

*Figure 1. EIP layers*
appears on the screens of network configuration tools.

Most importantly, your data is represented as generic data. The characteristics of your device are lost to the network. This option is only attractive if you expect low-volume requests for EIP connectivity.

**Add-on daughterboards**

An add-on daughterboard is another fast time-to-market method of adding EIP to your product. An add-on PCB plugs into your device internally, usually TTL-level communications, and acts as a serial gateway with less cost and sometimes more benefits.

Internal daughterboards contain all the disadvantages described previously for gateways unless the vendor is able to customize the data representation and messaging characteristics. If your product contains a spare communication port and you can get a customized implementation, this approach is great for medium volume applications (i.e., a few hundred units per year). Real Time Automation supplies daughterboards of this kind, not only for Ethernet, but also for other fieldbuses like Lonworks and DeviceNet.

**Design in an IC containing EIP**

Several companies are offering Microprocessor Units (MPUs) with EIP and other protocols embedded into the MPU. If you are building a new device from scratch and these MPUs fit your budget and taste, they can be a very attractive option. You control the object model implementation and messaging characteristics without the speed constraints imposed by gateways. It is also possible to create a similar solution by combining a System-On-a-Chip (SoC) with an EIP client/server or server-only stack as described below.

**Purchase an EIP client or server stack for your existing design**

When an off-the-shelf MPU with EIP is not within your budget or taste, you could purchase an EIP client/server or server-only stack. For a one-time charge with no royalties, these stacks offer the latest in capabilities while providing absolute control of development, network presentation, and implementation.

This approach requires a dedicated resource to integrate the MPU, TCP/IP stack, EIP stacks, and OS, if needed.

The advantages to this approach are relatively low cost and tightly integrated firmware. Disadvantages include the cost of dedicating internal resources to this effort and the resulting time-to-market issues.

**Porting code: The easy way vs. the hard way**

Real Time Automation made a very strategic move in implementing our EIP stack; we made it single-threaded, instead of multi-threaded. For all except the most demanding of applications, a single-threaded stack provides more than adequate performance with significantly less implementation effort. With its inherent simplicity, it can be easily ported to almost any RTOS or microprocessor architecture. Why spend months porting code and extending your time to market if you don’t really need to?

**Custom PCBs**

Contract engineering companies can provide a completely custom communications card for an application, usually in 90 days or less depending on the complexity of your application.

**Do it yourself**

The most costly, lengthy, and risky approach is to form an internal effort to build it yourself. While this is admittedly what you’d expect to hear from a company whose business is selling custom networking hardware and software, the facts still speak for themselves.

Like all complex protocol implementations, there are nuances to the EIP specification that are not readily discernable. For example, the specification provides reset functionality to the identify object of an EIP device. The nuances of how and when your device resets, and how this reset affects your client and server connections are very important. However, these details are beyond the scope of the EIP specification.

The internal resources required for an internal build are usually much more costly than using an outside resource or buying a component. The risk of missing seemingly innocuous nuances and missing a ship date is much higher than the cost of outside software or engineering.

**Your best customer – the guinea pig**

Anyone who has practiced engineering for any length of time has a great respect for Murphy’s Law. On the subject of networking, Murphy’s Law states that when you roll out a new network design, the odds are that you’ll have the opportunity to demonstrate your proficiency. You’ll demonstrate it on-site, in front of your biggest and most important customer as they wait impatiently for their network to return to life.

**Just a few more things to think about before you take the leap**

Another important consideration is testing and certification. Do you have the tools and resources to test, troubleshoot, and certify your EIP implementation? A key to selecting a vendor to get EIP in 90 days is to make sure that your vendor has all the right tools and can assist you with troubleshooting, locating adequate test tools, and certifying your device.

**You must consider documentation**

There are standard documentation formats for CIP devices. Your customers will expect your products to follow these standards. To get to market in 90 days, documentation issues need to be addressed early in the process.
You must consider certification
Many customers simply will not purchase products unless they have passed ODVA’s conformance test at the University of Michigan. It’s not uncommon for products to be sent back to the lab a second or third time before they finally pass. RTA was the first company to submit a product to the EIP conformance lab, and it passed the first time with flying colors. If RTA does an EIP implementation for your product, you are guaranteed the same first-time success.

You must consider maintenance and support
There are keys to success for any new technology. EIP is very new. There will be inevitable corrections and revisions to the initial specifications. How you deal with these changes matters a great deal.

You must consider factory floor EIP issues
What to do when devices have duplicate IDs, what to do when supposedly compatible devices still won’t talk to each other, how to use switches and routers on the factory floor, and how to handle device replacement are all very important factors to carefully consider.

Nonetheless, don’t let a shaky economy, protocol confusion, or fear of the unknown keep you from participating in new project opportunities. EIP can be an important part of your product success story, too.