Quick Start for Vendors Handbook

A Guide for EtherNet/IP™ Developers

www.odva.org

ODVA, Inc.
4220 Varsity Drive, Suite A
Ann Arbor, MI 48108-5006 USA
Phone +1 734.975.8840
Fax +1 734.922.0027
E-mail odva@odva.org
Web www.odva.org

Table of Contents

For the Reader

What is the purpose of this guide?

You are implementing EtherNet/IP™. Where do you start? What are your options? What issues should you consider? What do you need to know about the protocol? How should you proceed with your development?

This guide gives basic answers to the above questions. It provides an overview of the steps needed to implement EtherNet/IP, and offers practical guidance to help lead you to a successful EtherNet/IP implementation.

Who should read it?

Development engineers, development managers, product managers, and marketing will benefit from reading this guide. You are not expected to be an expert in EtherNet/IP or the Common Industrial Protocol (CIP™).

Technology Overview

What is EtherNet/IP?

EtherNet/IP is the name given to the Common Industrial Protocol (CIP), as implemented over standard Ethernet (IEEE 802.3 and the TCP/IP protocol suite).

EtherNet/IP was introduced in 2001 and today is the most developed, proven and complete industrial Ethernet network solution available for manufacturing automation, with rapid growth as users seek to harness the advantages of open technologies and the internet. EtherNet/IP is a member of a family of networks that implements CIP at its upper layers (Figure 1).

EtherNet/IP and CIP are managed by ODVA. ODVA publishes *The EtherNet/IP™ Specification* and helps ensure compliance through conformance testing.

What is CIP?

The Common Industrial Protocol (CIP) is a media independent, connection-based, object-oriented protocol designed for automation applications. It encompasses a comprehensive set of communication services for automation applications: control, safety, synchronization, motion, configuration and information. It allows users to integrate these applications with enterprise-level Ethernet networks and the Internet. Supported by hundreds of vendors around the world and truly media-independent, CIP provides users with a unified communication architecture throughout the manufacturing enterprise. CIP allows users to benefit today from the many advantages of open networks while protecting their existing automation investments when upgrading in the future. CIP brings:

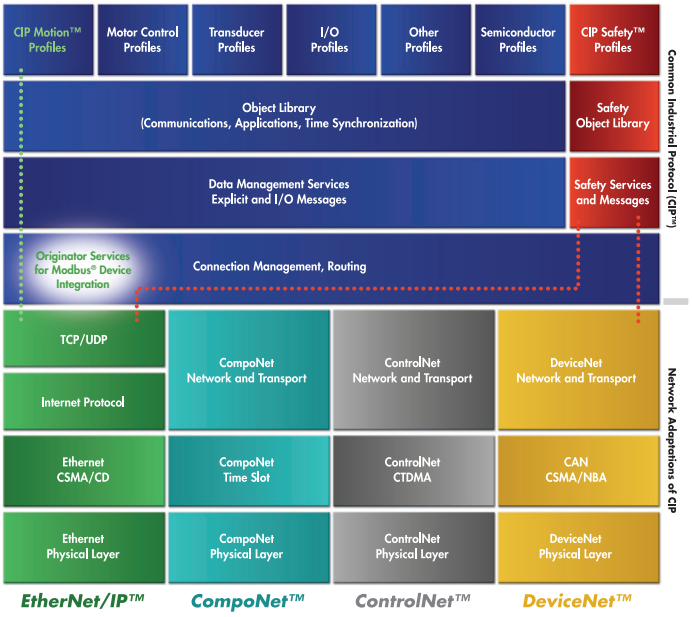


Figure 1
DeviceNet, CompoNet & ControlNet share the same CIP application layer with EtherNet/IP

- Coherent integration of I/O control, device configuration and data collection
- Seamless flow of information across multiple networks
- Ability to implement multi-layer networks without the added cost and complexity of bridges and proxies
- Minimized investment in system engineering, installation and commissioning
- Freedom to choose best of breed products, with the assurance of competitive prices and low integration cost

The "IP" in "EtherNet/IP" refers to "Industrial Protocol". EtherNet/IP utilizes CIP over standard IEEE 802.3 and the TCP/IP protocol suite (Figure 2). Since EtherNet/IP uses standard Ethernet and TCP/IP technologies, it allows compatibility and coexistence with other applications and protocols.

EtherNet/IP-enabled products have been developed using pre-existing hardware platforms, with existing TCP/IP stacks with multiple protocol support. A few vendors simply chose to provide a firmware update to an existing Modbus TCP interface, demonstrating the ease of integration and interoperability of CIP, without the need to develop specific hardware.

CIP and International Standards

CIP technologies are compliant with a number of fieldbus-related international standards, and are generally referred to as members of CPF 2 (Communication Profile Family 2) of IEC 61158.

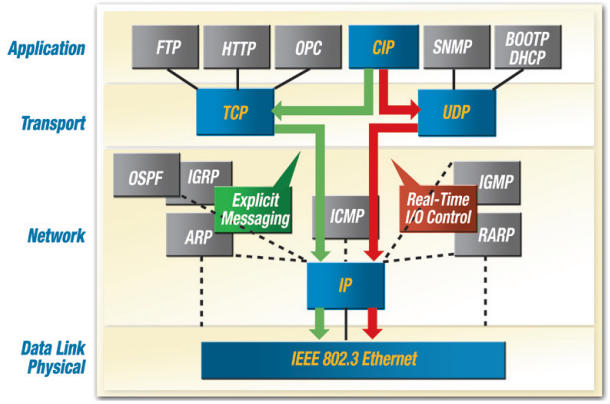


Figure 2
CIP is fully compatible with Ethernet and Internet protocols enabling multi-protocol support

- IEC 61158: Specifies various fieldbus protocols for applications ranging from discrete manufacturing to process control. It includes the specifications for CIP, as well as EtherNet/IP and ControlNet-specific protocol elements, as Type 2.
- IEC 62026-3: Specifies controller-device interfaces, and includes DeviceNet-specific protocol elements.
- IEC 61784-1 and IEC 61784-2: Specify general-purpose and real time Ethernet fieldbus Communication Profiles (CPs) (i.e., how to build a specific communication network using IEC 61158 and other standards). ControlNet, EtherNet/IP and DeviceNet are defined respectively as CP 2/1, CP 2/2 (CP 2/2.1 with CIP Sync), and CP 2/3.
- IEC 61784-3: Specifies Functional Safety Communication Profiles (FSCPs), i.e., extensions of fieldbusses for use in safety related applications. CIP Safety is included as FSCP 2/1.
- IEC 61918 & IEC 61784-5: Specify general and fieldbus-specific cabling installation guidelines. IEC 61784-5 includes specific guidelines for ControlNet, EtherNet/IP and DeviceNet.
- IEC 61800-7: Specifies profiles for power drive systems and their mapping to existing communication systems by use of a generic interface. It includes CIP Motion and its mapping on ControlNet, EtherNet/IP and DeviceNet.
- ISO 15745: Defines elements and rules for application integration, including communication network profiles and the communication aspects of device profiles for some fieldbus technologies. EDS files used for device and network integration of DeviceNet, ControlNet or EtherNet/IP applications are compliant with the relevant parts of ISO 15745 (respectively Parts 2, 3 and 4).

Also:

- The lower layers of DeviceNet are based on **ISO 11898**, also known as CAN (Controller Area Network).
- The lower layers of EtherNet/IP are based on the various RFC internet standards for the TCP/UDP/IP suite, on the IEEE 802.3 and ISO Ethernet standards (**ISO/IEC 8802-3**), without modification or extension (Figure 3).
- CIP Safety (on DeviceNet and EtherNet/IP) has been certified for use in applications in systems needing to meet the requirements of **IEC 61508** up to and including SIL3.

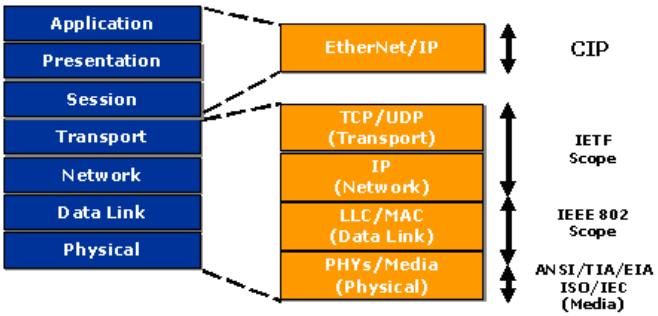


Figure 3
EtherNet/IP and the OSI model

Who is ODVA?

ODVA is an international association comprising members from the world's leading automation companies. Collectively, ODVA and its members support network technologies based on the Common Industrial Protocol (CIP™). These currently include DeviceNet™, EtherNet/IP™, CompoNet™, and ControlNet™, along with the major extensions to CIP — CIP Safety™, CIP Sync™ and CIP Motion™. ODVA manages the development of these open technologies, and assists manufacturers and users of CIP Networks through its activities in standards development, certification, vendor education and industry awareness.

As part of its certification activities, ODVA offers conformance testing to help ensure that products built to its specifications operate in multi-vendor systems. Developers are encouraged to contact ODVA early during the development process to obtain the necessary technical specifications and to become an authorized EtherNet/IP vendor. Developers are also recommended to take part in the Workshops for Implementors of EtherNet/IP and interoperability testing run at PlugFests by ODVA. Details of forthcoming events can be found on the ODVA home page: www.odva.org.

For further information on ODVA, see the ODVA website: www.odva.org

How is this guide related to *The EtherNet/IP Specification*?

As of November 2008, *The EtherNet/IP Specification* consists of three parts: *The Common Industrial Protocol* (Volume 1 of the CIP Networks Library), the *EtherNet/IP Adaptation of CIP* (Volume 2), and the *Integration of Modbus® Devices into a CIP Architecture* (Volume 7). The specifications contain all of the normative information for the protocol – meaning it contains every detail that is considered part of the standard. In contrast, this guide provides practical developer information not found in the specifications.

The information in this guide does not replace *The EtherNet/IP Specification.*

Objects, Services, and Application Data

What services does EtherNet/IP provide?

The CIP application layer defines a set of application objects and device profiles that define common interfaces and behaviors. In addition, CIP communication services enable end-to-end communication between devices on the different CIP networks. EtherNet/IP maps the CIP communication services to Ethernet and TCP/IP, enabling multi-vendor interoperability between devices on Ethernet as well as with the other CIP networks.

Simplified EtherNet/IP Object Model Overview

Within the CIP application layer, devices are represented using an object model (Figure 4). Application objects define how device data is represented and accessed in a common way. Network-specific objects define how parameters such as IP addresses are configured and EtherNet/IP specific functions.

Communication objects and services provide the means to establish communication associations and access device data and services over the network.

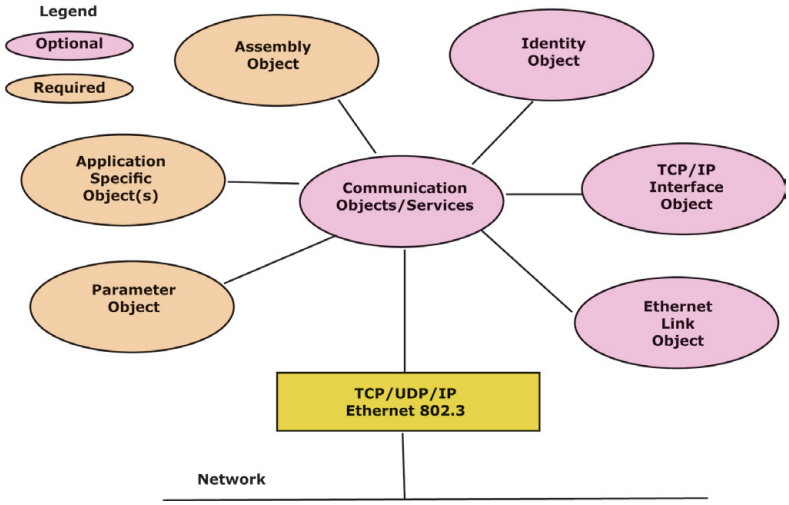


Figure 4
This EtherNet/IP simplified Object Model defines data, services, connections, and their relationships

Exposing Application Data with CIP

Objects within a device are groups of related data and behavior associated with this data. CIP requires certain objects to describe a device, how it functions, communicates and its unique identity. The Identity Object, for example, contains identity data values called attributes that are used to store the identity information of a device. Attributes for the Identity Object include the Vendor ID, Device Type, device serial number and other identity data. CIP does not specify how object data is implemented, rather, which data values or attributes must be supported and made available to other CIP devices.

There are three types of objects defined by CIP:

Required Objects must be included in all CIP devices. These objects include the Identity Object, the Message Router Object and network-specific objects.

Application Objects describe how data is encapsulated by a device. These objects are specific to the Device Type and function. For example, an input device would have an input object with attributes that describe the value and fault status of a particular input point.

Vendor-specific Objects describe services that are specific to a particular vendor; they are optional and not described in a predefined Device Profile.

Addressing data within a CIP device utilizes the same object-oriented view. A class (of objects) is a set of objects that represent the same type of system component (Figure 5). Sometimes it is necessary to have more than one 'copy' of an object, called object instances, within a device. This set of objects is called an object class. Each instance of the object class will have the same set of attributes, but will have a unique set of values. An object instance or an object class has attributes, providing services and implementing behavior.

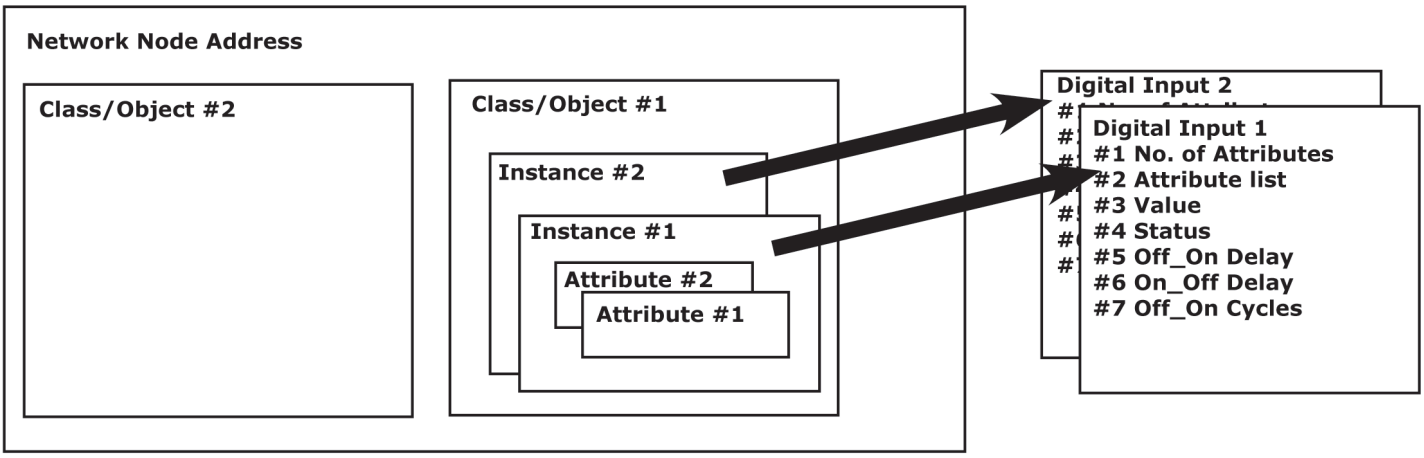


Figure 5
CIP Object-oriented view of application data

Accessing data within a device using a non-time critical message (an explicit message – see *Types of EtherNet/IP communications* later) typically contains the following address information:

- Device network address
- Class ID
- Instance ID
- Attribute ID
- Service code (describing the action/service required)

The Class/Instance/Attribute ID form of addressing is also used in Electronic Data Sheets (EDS) to identify configurable parameters within a device.

In addition to specifying how device data is represented, CIP also specifies methods by which I/O data can be accessed, using triggers, such as cyclic or change-of-state. Vendors can also describe how data from different objects can be combined in an I/O or configuration message using the Assembly Object.

Implementation Approaches

Steps to EtherNet/IP implementation

While each implementation will have its own specific requirements, the following is a list of the general steps most implementations will follow:

- Understand and define your application and user requirements
- Determine the level of EtherNet/IP capability you require – what type of device and what type of communication services
- Select an implementation approach – build or buy, hardware or software
- Develop the EtherNet/IP implementation
- Develop an EDS that describes the device capabilities
- Test your implementation
- Have the product conformance tested

The following sections of the guide discuss these topics in more detail.

Getting started with EtherNet/IP

The starting point for any successful development effort, EtherNet/IP or otherwise, is to understand your application and customer requirements. There are several issues you should consider:

- What are your primary reasons for implementing EtherNet/IP?
- In what application(s) will your implementation be used?
- Do you need to meet specific regulatory or industry standards?
- With which other devices of applications must you communicate?
- What are the characteristics of the communications (which data, how large, how frequent)?
- Do you need to support other protocols in addition to EtherNet/IP?
- What are the performance requirements?
- What are the time-to-market and cost requirements?

The answers to these questions will help you select the right path to implementation.

In addition, in order to become an ODVA-authorized EtherNet/IP vendor, you will need to execute and comply with an ODVA Terms of Usage Agreement. View suggestions for getting started on the Quick Start link on www.odva.org for more information about becoming an EtherNet/IP vendor. Also see the EtherNet/IP Development Checklist in this document.

Selecting the right level of EtherNet/IP device capability

With EtherNet/IP you can create a simple device such as a digital I/O module, or a complex device such as a programmable controller. Devices may have differing levels of communications capability.

How do you know what level of EtherNet/IP capability you need? Your application requirements should determine the level of capability that is suitable:

- With which specific devices or applications must you communicate?
- What are the capabilities of the devices or applications with which you must communicate?
- Which device initiates the communications?
- How much data is exchanged?
- How frequently is data exchanged?
- What are the communication latency expectations?
- What is the desired user experience with your device (e.g., configuration, application integration and diagnostics)?

Types of EtherNet/IP communications

EtherNet/IP defines two primary types of communications: explicit and implicit (Table 1).

Table 1

CIP Message Type	CIP Communication Relationship	Transport Protocol	Communication Type	Typical Use	Example
Explicit	Connected or Unconnected	TCP/IP	Request/reply transactions	Non time-critical information data	Read/Write configuration parameters
Implicit	Connected	UDP/IP	I/O data transfers	Real-time I/O data	Real-time control data from a remote I/O device

Explicit Messaging in general has a request/reply (or client/server) nature. This type of communication is used for non-real-time data, normally for information. Explicit messages include a description of their meaning (expressed explicitly), so the transmission is less efficient, but very flexible. It may be used by an HMI to collect data, or by a device programming tool. In CIP terms, with Explicit Messaging you request a service of a particular object, e.g., a read or a write service. For EtherNet/IP, Explicit Messaging uses TCP. Explicit Messaging can be done with or without prior establishment of a CIP connection.

Implicit Messaging is also often referred to as "I/O" and is time-critical in nature. Typically this type of communication is used for real-time data exchange, where speed and low latency are important. Implicit messages include very little information about their meaning, so the transmission is more efficient, but less flexible than explicit. The interpretation of the transmitted data is fast. With Implicit Messaging you establish an association (a "CIP connection") between two devices and produce the Implicit Messages according to a predetermined trigger mechanism, typically at a specified packet rate. The devices both know and agree on the data formats they will use (i.e., the format is "implied"). For EtherNet/IP, Implicit Messaging uses UDP and can be multicast or unicast.

Connections are established using the ForwardOpen Request service of the Connection Manager Object. The ForwardOpen Request contains all of the connection parameters, including transport class, production trigger, timing information, electronic key and connection IDs. Connection clean-up takes place when a ForwardClose Request service request is issued or when either connection end point times out.

Implicit messaging can make use of the CIP Producer/Consumer communication model. With Producer/Consumer, the producing device transmits data once, regardless of the number of consumers. All interested consuming devices receive the same data. For EtherNet/IP the produced data is identified by the IP multicast address and the CIP Connection ID. The Producer/Consumer model leads to greater network efficiency when multiple consumers need to receive the same data from a producer. For I/O connections, once the connection is established there is no request/response, the data with the ConnectionID is just produced and consumed at intervals determined by the Production Trigger which was specified at connection establishment. Triggers can be Cyclic (most common), Change of State (CoS) or Application.

Types of EtherNet/IP devices

Several device classifications, based on their general behavior and types of EtherNet/IP communications they support, have been defined by the participants in the Workshops for Implementors of EtherNet/IP in order of the most simple to the most complex:

Explicit Message Server: An explicit message server responds to request/response oriented communications initiated by explicit message clients. An example of an explicit message server is a bar code reader.

Explicit Message Client: An explicit message client initiates request/response oriented communications with other devices. Message rates and latency requirements are typically not too demanding. Examples of explicit message clients are HMI devices, programming tools, or PC or Linux based applications that gather data from control devices.

I/O Adapter: An I/O adapter receives implicit communication connection requests from an I/O scanner then produces its I/O data at the requested rate. An I/O adapter is also an explicit message server. An I/O adapter can be a simple digital input device, or something more complex such as a modular pneumatic valve system.

I/O Scanner: An I/O scanner initiates implicit communications with I/O adapter devices. A scanner is typically the most complex type of EtherNet/IP device, as it must deal with issues such as configuration of which connections to make, and how to configure the adapter device. Scanners also typically support initiating explicit messages. A programmable controller is an example of an I/O scanner (Figure 6).

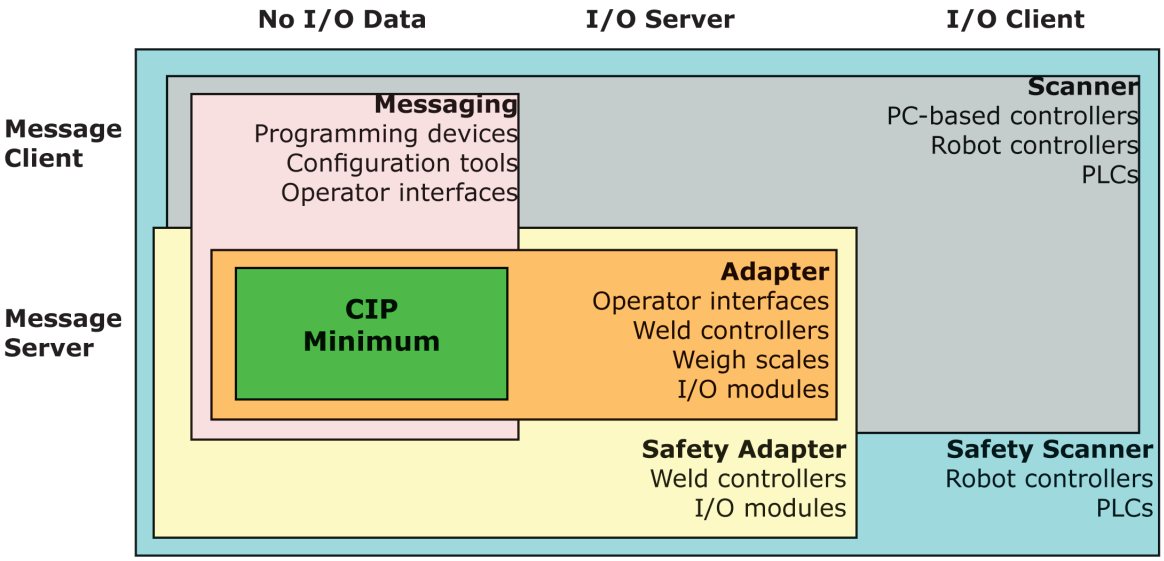


Figure 6
Matrix of communications types and device classification

How do you know which type of device, and which type of communications you need? You should first note that all EtherNet/IP devices are required to have minimal explicit server capability, in order to respond to device identification and configuration requests.

You should also consider the other devices with which you must communicate, and what type of communications they support. If you must communicate with a device that only supports explicit message server communications, then you need an explicit message client.

The nature of the communications should also help you determine whether explicit or implicit communications are most suitable. Explicit messaging, while being generally easier to implement, is more suited to lower performance, request/response types of communications. Implicit messaging, while being more complex to implement, is needed when higher performance and more deterministic communications are needed.

Another important consideration is how the type of communications affects the interface that will be exposed to the user. For example, I/O scanner devices often have different interfaces for implicit and explicit messaging. You should consider which interface is best suited to the user of your device or application.

Additional device considerations

Beyond the core EtherNet/IP communications capability, you should consider several additional features for your implementation:

Ethernet physical layer: *The EtherNet/IP Specification* (Volume 2, Chapter 8) defines requirements for an industrial-grade physical layer design. The specification defines environmental requirements, cabling, connectors, and other factors. Note the industrial physical layer is not required (but highly recommended) of EtherNet/IP devices. You should consider whether this is needed by your application environment.

Linear topology: Some users want an alternative topology to the traditional Ethernet switched star. A linear topology has advantages for certain applications (such as conveyors). Some users may also have a preference for the traditional fieldbus topology, simplifying network cabling and eliminating the cost of switches. This can be achieved with a device having 2 Ethernet ports with embedded switch technology or by using a simple 3 port switch to connect single port devices.

Indicators and diagnostics: Diagnostic indicators are a good tool for application troubleshooting, and should be considered where feasible. *The EtherNet/IP Specification* (Volume 2, Chapter 9) also defines common indicator labeling and behavior for EtherNet/IP devices. If you have indicators on your device, you should follow the behavior in the EtherNet/IP specification. The module status and network status LEDs are required if you support the industrial physical layer.

Performance: Good performance, relative to the application, is a characteristic of a good implementation. Performance will depend on several factors, including processor speed and the efficiency of your implementation. It is particularly important to consider the performance of your TCP/IP stack, since different stacks vary widely in their performance. You should ask the vendor of your TCP/IP stack (often included with the operating system) for performance information. If you are purchasing an EtherNet/IP implementation (hardware or software), you should also make sure its performance will meet your application requirements (*see Further Information: Recommended Functionality for EtherNet/IP Devices*).

Security: The increasing size, variety, and connectivity of industrial networks requires increasing levels of device robustness. Consider the following as a minimum:

Port Scanning - nmap (www.insecure.org)
Known Flaw Checks - Nessus (www.nessus.org)
Resource Starvation - Broadcast/multicast overload
Protocol Robustness - Malformed packets

Optional EtherNet/IP features: Implementing optional EtherNet/IP features not mandated by the specification can help to ensure better interoperability, creating a better user experience. For example, implementing the IP address conflict detection mechanism (ACD) allows a device to detect that another device is using its IP address. The most important features for good interoperability are defined in the ODVA document *Recommended Functionality for EtherNet/IP Devices* (see *Further Information*).

Additional protocols: Do you need to support additional TCP/IP protocols? Many devices have found an embedded web server to be valuable mechanism to provide diagnostic information. DHCP is the recommended mechanism for initial IP address configuration out-of-box. You should consider additional protocol support when evaluating development platforms. It is necessary to ensure that prioritization within the node assumes that handling of the additional protocols (such as HTTP or SNMP) does not interfere with the prompt handling of EtherNet/IP I/O connections.

Modbus®: The library of specifications for CIP Networks has been enhanced to provide seamless access to Modbus devices. This is achieved by implementing CIP-Modbus translation defined in Volume 7 of the CIP Networks Library: *Integration of Modbus Devices into the CIP Architecture*. This enables existing unmodified Modbus devices to be seamlessly accessed from CIP originator devices. Use of the CIP-Modbus translator requires no changes for Modbus target devices. Typical candidate devices include:

- CIP originators - Implement the Modbus functionality in their device, or allow their devices to communicate via a CIP-to-Modbus router by providing addressing capabilities for offlink targets.
- CIP routers - Implement the Modbus functionality in a CIP-to-Modbus router (e.g., an EtherNet/IP to Modbus Serial router).
- Modbus devices – Facilitate CIP-to-Modbus integration by creating EDS files that represent the addressable data items in your device.

What are the different ways to implement EtherNet/IP?

There are several different ways to implement EtherNet/IP: software or hardware, develop your own, or purchase all or part of an implementation.

Hardware

- Add interface daughter card to an implementation
- Microprocessor with embedded EtherNet/IP
- Complete custom hardware
- External protocol gateway device (e.g., serial to EtherNet/IP)

Software

- Integrate an EtherNet/IP stack into an implementation
- Write your own EtherNet/IP stack
- Complete outsourced custom development

How do you determine which of these approaches is the most suitable? As mentioned earlier, understanding your specific reasons and requirements for implementing EtherNet/IP is a good starting point. In addition you should consider your background and experience in several areas that may be relevant to your project:

- Hardware development
- Ethernet communications
- TCP/IP protocol suite
- Embedded device development
- CIP and EtherNet/IP knowledge

If you are unfamiliar in any of these areas, there are a number of companies that can offer assistance, training and a variety of implementation approaches. For a list, see the ODVA website: www.odva.org.

Also consider time-to-market requirements, development budget and product cost targets. Will you have the resources to support and maintain the product?

Each of the different approaches has advantages and disadvantages and is appropriate in certain situations (Table 2).

Table 2

Approach	Advantages	Disadvantages	Best suited to
Gateway interface or external protocol gateway This approach utilizes separate hardware that is external to the device and communicates with the device using a serial protocol, converting the serial protocol to EtherNet/IP.	• Quickest path to implementation • No need to do any actual EtherNet/IP implementation	• Software tools see external gateway as the EtherNet/IP device • User must configure the gateway device to obtain end device data • Limited EtherNet/IP communication options (size, rate, type) • Lowest communications performance	Low volume, low performance situations that need a fast solution.
Embedded board/module with EtherNet/IP implementation	• Ethernet hardware development already done • EtherNet/IP stack already done • Can be customized to give unique vendor identity • Performance acceptable for most devices	• Cost of the interface card • Functionality is limited to the features already incorporated into the design of the daughter card • Performance can be an issue in the most demanding applications	Low or mid volume situations, or when connectivity to multiple fieldbusses is needed.
Microprocessor with embedded EtherNet/IP	• EtherNet/IP stack already developed • Often can also use a corresponding reference hardware platform • Provides flexibility on the functionality and features that can be offered in your product • Performance can be very good (depending on hardware) • Can be customized to give unique vendor identity	• Hardware development to integrate with chosen platform • Cost can be an issue versus an off-the-shelf processor	Higher volume or higher performance situations when the vendor does not wish to do complete hardware development or integrate an EtherNet/IP stack.
Complete custom hardware development (in-house or out-sourced)	• Can choose the ideal processor and components • Total control on functionality and features that can be offered in your product • Can achieve high performance • More control over cost • Can choose the appropriate OS, TCP/IP stack, EtherNet/IP stack	• Most work to develop	Vendors with hardware development expertise, or when product cost is critical (e.g., high volume).

Approach	Advantages	Disadvantages	Best suited to
Integrate an EtherNet/IP stack	• Don't need to do the stack development • Able to choose a stack that meets your needs	• Still may need to develop a hardware platform • Still may need to select a TCP/IP stack	Applications running on existing hardware, or when you are comfortable developing the hardware and need an EtherNet/IP stack.
Implement an EtherNet/IP stack	• Able to implement the EtherNet/IP aspects that you need • More control over performance, quality, maintenance	• Significant work for a full-featured stack • Likely to have errors in initial versions	Simple explicit message clients and servers, especially running on Windows or Linux (or similar) systems; EtherNet/IP stack providers; larger vendors who must re-use across multiple products.

If you are integrating an EtherNet/IP hardware or software solution from another vendor, look for solutions that are conformance tested or already used in other compliant products; ideally, the implementation should have also passed interoperability testing via an ODVA-sponsored PlugFest. Consider vendors who are members of ODVA. These companies have made a commitment to EtherNet/IP; some are involved in EtherNet/IP technology development, seminars or workshops, and they will have significant experience with EtherNet/IP. Find them at www.odva.org.

Developing Your Implementation

What should you know about EtherNet/IP?

What you need to know about EtherNet/IP depends on what type of development you are undertaking. If you are writing your own I/O adapter or I/O scanner stack, then you should be quite familiar with *The EtherNet/IP Specification*.

If you are writing an explicit message client or server stack, you should be familiar with the specification aspects regarding explicit communications as well as the appropriate application objects.

If you are integrating an EtherNet/IP hardware or software solution into your existing implementation, you need to know enough to integrate and customize the solution for your unique situation.

All developers need to know enough about EtherNet/IP to be able to debug their implementation and support their users in the field.

Understanding the following essential EtherNet/IP topics will assist your implementation. Please see the section on *Further Information* for references to additional material on these essential topics.

Basics of CIP Communications: Basic concepts of Explicit and Implicit (I/O) communications; connected and unconnected messages; the Forward Open service and connection establishment; types of I/O application connections. You should be able to understand the basic sequence of message transactions that occur between two EtherNet/IP devices. Appendix C provides message flow diagrams to help with this understanding. You should understand the Run/Idle notification mechanism of I/O connections and its use in control applications. Controllers/scanners typically use it to notify their devices that they have gone into program (idle) mode. It is important that target devices react to this notification appropriately.

CIP Object Model: Basics of the CIP object model including object classes, instances, attributes, services.

CIP Object Library: You should be familiar with the Identity Object. If you are implementing an I/O scanner or adapter you should be familiar with the Assembly Object. EtherNet/IP embedded devices must implement the TCP/IP Interface Object and Ethernet Link Object for network configuration.

EtherNet/IP Encapsulation Layer: Basic concepts of the encapsulation layer used to transport CIP over TCP and UDP including where TCP and UDP are used; EtherNet/IP registered port numbers; usage of IP multicast for I/O connections.

Device Profiles: Which device profile best fits your implementation? Refer to the section on *Select a device profile.*

Physical layer requirements: If you need the industrial physical layer, what are the physical layer requirements? The best resource here is *The EtherNet/IP Specification* (Volume 2, Chapter 8).

Electronic Data Sheet (EDS): How to create an EDS file for your device. The basic concepts of an EDS should be understood (see Appendix A)

Practical aspects of implementing EtherNet/IP are the subject of Workshops for Implementors of EtherNet/IP, open to all developers. See www.odva.org for dates and locations. The resulting guidance is incorporated into several recommendation documents available from ODVA.

Developing your EtherNet/IP Implementation

After you have determined your application requirements, selected an implementation approach, and learned what you need to know about EtherNet/IP, you will begin your development. While this guide cannot describe how to develop your own product, there are a number of important EtherNet/IP aspects you should address:

Select a device profile. Profiles define the required objects and communication formats for devices. You should select a profile appropriate to your device type. Specific profiles are for devices that share the same behavior have the same I/O data and configuration attributes. Many device profiles are already defined in library of specifications for CIP Networks, but you may have a device that doesn't fit an existing profile. If an existing profile is not appropriate, ODVA Members have the option of creating one via the ODVA specification enhancement process. You can also use the Generic Device profile, which provides minimum functionality. An alternative is to develop your own: you may adopt or develop a device profile for your product, known as a 'Vendor-specific' profile, which will need to have the same required objects mentioned earlier.

Select a device identity. Device identity information is needed for EtherNet/IP configuration tools. Identity information is stored in the Identity Object and consists of the Vendor ID, Device Type, Product Code and Major Revision for your device. The Vendor ID is assigned by ODVA and may be obtained via the order form on www.odva.org. The Device Type reflects the device profile, while the Product Code and revision information is chosen by the vendor of the product.

Interoperability with other devices. You should understand the capabilities and points of interoperability with the other devices or applications with which you must communicate. Make sure you have considered any issues with byte ordering or data alignment. Also make sure you support the specific communication parameters required by the other devices (e.g., accepting electronic key segments, producing multicast packets, etc.). ODVA recommendations documents provide guidance to ensure interoperability among devices (listed in *Further Information*).

Define the application data. An important step in the development process is to define your application data. What data must be exchanged with the communication partners? In which CIP application objects will the data be represented? For most EtherNet/IP I/O adapters, the data will be available via the Assembly Object. If you use a profile other than a Generic Device, additional application objects may be required. You can also define a vendor-specific application object, but be aware that for the greatest interoperability, you should use a standard object.

Integrate the EtherNet/IP stack or hardware. If you are using an EtherNet/IP stack or hardware solution from another vendor, you will need to integrate it with your device or application functions. The vendor of the EtherNet/IP stack or hardware should be able to help you with the integration points.

Create an EDS file. An Electronic Data Sheet (EDS) is an ASCII text file that describes the features of EtherNet/IP device and is used by software tools for device and network configuration. As a minimum, an EDS needs to contain identity information to allow a tool to recognize the device. While this is sufficient for messaging servers and clients in most cases, scanners and adapters need to have their connection capabilities described. For a typical I/O Messaging Adapter, it is sufficient to create one entry for every I/O Connection that can be made to the device. Adding more descriptive details, such as the elements of the I/O and Configuration Assemblies, are necessary to help the user understand the I/O data and configure the device.

An EDS file can be placed in a device using the File Object in its native form or in a compressed (zipped) file. The advantage of this approach is that the user always has access to the correct version of the EDS for the version of the product held. See Appendix A for more information on creating an EDS file.

Publish a user manual. Do include the Assembly Object definition and assembly instances. (These will be in the EDS file too.)

Testing Your Implementation

Testing your implementation

If you want to ensure the best possible quality and experience for your users, you should test your EtherNet/IP implementation in several areas.

In-house testing. You should test your implementation in systems and scenarios that are relevant to the application. If possible, you should test for interoperability with devices or software your users will use. Consider doing exception testing, such as network cable disconnects and operation with high network loads, since problems are often found in these scenarios.

EtherNet/IP Conformance Testing is mandatory according to the ODVA Terms of Usage Agreement. Vendors are required to maintain conformance and correct any non-conformances should they occur. Testing is undertaken at an ODVA-authorized Test Service Provider; centers are located in Europe, the USA and Asia. Devices that pass conformance testing are eligible to receive a Declaration of Conformity (DOC). DOCs issued are published by ODVA on www.odva.org.

Figure 7
Certification Mark for Products Holding a Declaration of Conformity for an EtherNet/IP Device

EtherNet/IP Interoperability Testing. The series of Workshops for Implementors of EtherNet/IP have defined a set of interoperability recommendations (see *Further Information*) to help ensure interoperability among devices. These recommendations are tested at ODVA-sponsored events called "PlugFests" that occur approximately twice per year, once each in North America and Europe. The interoperability tests are optional *(advisory)*, but have been proven to be highly valuable in creating interoperable products that work in user applications. Developers have found early participation during the development process to be most beneficial. Interoperability testing also includes security testing. Elements of the current advisory test *may* become part of the standard conformance test.

Some users are now insisting on product that has passed both conformance and interoperability testing. Performance testing and results are also becoming of interest to users, and are likely to be utilized as selection criteria.

See www.odva.org for the schedule of Workshops for Implementors and PlugFests for EtherNet/IP. **Note:** Products should obtain a Declaration of Conformity before participating in a PlugFest or are required to pass Conformance Test using the same firmware within 6 months after the PlugFest in order to be eligible for the Interoperability Advisory Mark to be added to the Declaration of Conformity. Vendors are encouraged to participate during the development cycle.

Performance testing. ODVA offers optional device performance testing. It is recommended to undertake performance testing during the product development cycle. Understandably, users are showing interest in performance testing in the selection of EtherNet/IP components. Contact ODVA for further details.

Development Tools

What development tools are available?

Set up an example EtherNet/IP network. Ideally you should set up an example EtherNet/IP network with devices with which you must communicate. You can then more easily understand EtherNet/IP from the end-user perspective and learn about EtherNet/IP communications. Later the network can be used for functional and interoperability testing.

If you are developing an I/O scanner or adapter, you would set up a network with a scanner (e.g., logic controller) and an I/O adapter. If you are developing an explicit message client or server, you will need those devices. Often I/O scanners can also function as explicit message clients.

A switch (with port monitoring/port mirroring) or hub will be necessary to monitor traffic.

For EtherNet/IP product information, see ODVA's CIP Supplier Directory at www.odva.org or contact a vendor for a specific recommendation. Ideally, base your network on some of the same devices your customer will use.

Protocol analyzers

A protocol analyzer with CIP decoding[1] provides an excellent environment to learn CIP messaging and for trouble shooting device or network communications. Most protocol analyzers can perform live traffic capture, offline analysis and usually provide some level of packet filtering.

A managed switch with port mirroring will be required to see all the network traffic. Older hubs may also be used (although these are becoming increasing hard to find), however, ensure it is not a switched hub.

Other Development Tools

- **EZ-EDS** is a freeware tool for Electronic Data Sheet (EDS) creation and maintenance. EZ-EDS reduces the time to create an EDS for a specific product and provides support for all EDS constructs. Available from: www.odva.org
- **Protocol Conformance Test Software Tool** enables self-testing of EtherNet/IP devices prior to attending a PlugFest and prior to having the test performed by an ODVA Test Service Provider. Vendors have found the tool useful to check device functionality during development. Available from: www.odva.org

ODVA Membership: Developer Benefits

By joining ODVA, developers can obtain various benefits, including the ability to work in collaboration on enhancements to the specifications, preferential pricing on classes, seminars and workshops run by ODVA, and the eligibility to review a prepublication copy of the specifications. Software and services utilized for development and testing also qualify for preferential rates. Other membership benefits include product promotion and marketing through the web, catalogs, trade show participation and Territory Alliance Groups.

ODVA Members implementing EtherNet/IP may also wish to join an ODVA Special Interest Group (SIG). These groups work to develop, monitor, and approve specification enhancements. SIGs are active in a number of areas, such as physical media, software tools, system architecture, conformance, device-specific (e.g., drives), and vertical markets (e.g., automotive and semiconductor).

---

1 Some available protocol analyzers and other development tools used in the EtherNet/IP PlugFests are listed in Appendix D.

Further Information

ODVA Publications

The following documents are available from www.odva.org under the EtherNet/IP Library unless otherwise indicated. If an ODVA publication number (PUB) is assigned, it is indicated after the document's title.

To visit the EtherNet/IP Library, go to www.odva.org, and from the home page, click:
'ODVA Technologies' → *'EtherNet/IP'* → *'EtherNet/IP Library'*

ODVA Terms of Usage Agreement
Outlines the requirements for using an ODVA technology and must be executed to obtain *The EtherNet/IP Specification* and become an EtherNet/IP vendor.

The EtherNet/IP Specification
The specification contains all of the normative information for the EtherNet/IP protocol. Available by subscription. Order forms can be found on www.odva.org.

The CIP Advantage Technology Overview Series: EtherNet/IP **(***ODVA PUB00138)*
A short technical introduction to EtherNet/IP.

General Recommendations for EtherNet/IP Developers *(ODVA PUB00100)*
Assists vendors to develop EtherNet/IP products that best meet the needs of end users. Many of these recommendations are based upon experiences from early adopters and EtherNet/IP technology providers.

IPv4 Address Conflict Detection for EtherNet/IP *(ODVA PUB00127)*
Recommends a common mechanism by which EtherNet/IP devices can detect IP address conflicts.
PlugFest requirement

Recommended Functionality for EtherNet/IP Devices *(ODVA PUB00070)*
Recommends functionality to be implemented to ensure interoperability between devices and provide a reasonable minimum level of capability above and beyond the minimum required for compliance.
Includes PlugFest requirements

Recommended IP Addressing Methods for EtherNet/IP Devices *(ODVA PUB00028)*
Recommendations for common mechanisms related to IP address assignment and management for EtherNet/IP devices.
PlugFest requirement

EtherNet/IP Interoperability Test Procedures *(ODVA PUB00095)*
Describes the eligibility of candidate devices and the interoperability testing for EtherNet/IP devices.
PlugFest Test Procedure

Network Infrastructure for EtherNet/IP *(ODVA PUB00035)*
Provides recommendations on infrastructure decisions.

EtherNet/IP Media Planning and Installation Guide *(ODVA PUB00148)*
Describes the required media components, how to plan, install, verify, troubleshoot, and certify an EtherNet/IP network. *www.odva.org*

The Common Industrial Protocol and the Family of CIP Networks *(ODVA PUB00123)*
A detailed technical description of the Common Industrial Protocol and its networks.

Other ODVA Resources

Ask the Experts
Q&A forum staffed by a panel of ODVA staff and volunteer member EtherNet/IP experts who answer technical questions or provide information on EtherNet/IP.

EtherNet/IP Quick Start for Vendors
A one-day training course designed to assist developers accelerate implementation. The latest schedule of events can be found on www.odva.org.

Workshops for Implementors of EtherNet/IP
The workshops educate product developers on EtherNet/IP and Industrial Ethernet technologies, provide a collaborative forum for developers to discuss common issues and solutions, and promote EtherNet/IP device interoperability through product design recommendations and an interoperability validation event (PlugFest). The latest schedule of events can be found on www.odva.org.

Past workshop presentations include:

- Essential CIP and EtherNet/IP for Developers
- Ethernet, TCP/IP and CIP
- EtherNet/IP Performance Guidelines
- Modbus/TCP Integration Overview
- Quality of Service for EtherNet/IP
- Understanding Terms of Usage and Declarations of Conformity
- Security Considerations & Securing EtherNet/IP Networks

Glossary

Adapter Class Device
An Adapter Class product emulates functions provided by traditional rack-adapter products. This type of node exchanges real-time I/O data with a Scanner Class product. It does not initiate connections on its own (see I/O Adapter).

Application I/O Trigger
The Application Trigger is one of three types of I/O triggers supported by CIP for the exchange of data on I/O connections. It is very similar to the CoS trigger and not common.

Broadcast
A broadcast transmission is a packet that all nodes on the network receive.

Encapsulation Protocol
Defines the communication relationship between two nodes known as an Encapsulation Session. The Encapsulation Protocol uses TCP/UDP Port 44818 for several Encapsulation Commands and for CIP Explicit Messaging. An example encapsulation command is the List_Identity Command that performs a "network who". An Encapsulation Session must be established before any CIP communications can take place. Data format for the Encapsulation Protocol is *Little-Endian.*

Change of State I/O Trigger
Change of State (CoS) is one of three types of I/O triggers supported by CIP for the exchange of data on Class 0 or 1 I/O connections. CoS endpoints send their messages when a change occurs. The data is also send at a background cyclic interval if no change occurs to keep the connection from timing out.

Connection Establishment/Close
Connections are established Connection Originators using the ForwardOpen service and closed by using the ForwardClose service. Connection clean-up takes place when either connection end point times out.

Connected Messaging
A CIP connection is a relationship between two or more application objects on different nodes. The connection establishes a virtual circuit between end points for transfer of data. Node resources are reserved in advance of data transfer and are dedicated and always available. Connected messaging reduces data handling of messages in the node. Connected messages can be Implicit (I/O) or Explicit.

Connection Originator
The source node that makes a request to a Connection Target for a connection. It can initiate either an I/O connection or explicit message connection using the ForwardOpen service.

Connection Target
Destination for I/O or explicit message connection requests. Responds to a connection request with a ForwardOpen service response.

Cyclic I/O Trigger
Cyclic is one of three types of I/O triggers supported by CIP for the exchange of data on Class 0 or 1 I/O connections. Endpoints send their messages at pre-determined cyclic time intervals.

EDS File
An Electronic Data Sheet (EDS) is an ASCII text file that describes the features of an EtherNet/IP device and is used by software tools for device and network connection configuration.

Exclusive Owner Connection
This is one of three types of Implicit (I/O) Connections. It is a Class 0 or 1 bidirectional connection to an Output connection point (typically an Assembly Object), where the data of this assembly can only be controlled by one Scanner. There may be a connection to an input assembly; this data is being sent to the scanner. If the input data length is zero, then this direction becomes a Heartbeat connection.

Explicit Messaging
Explicit Messages can be sent as a connected or unconnected message. CIP defines an Explicit Messaging protocol that states the meaning of the message. This messaging protocol is contained in the message data. Explicit Messaging provide the means by which typical request/response oriented functions are performed (e.g., module configuration). These messages are typically point-to-point. Message rates and latency requirements are typically not as demanding as I/O messaging.

Explicit Message Client
An explicit message client initiates request/response oriented communications with other devices. Examples of explicit message clients are HMI devices, programming tools, or PC or Linux based applications that gather data from control devices.

Explicit Message Server
An explicit message server responds to request/response oriented communications initiated by explicit message clients. An example of an explicit message server is a bar code reader.

ForwardOpen Service Request
The ForwardOpen Service Request is sent by the Connection Originator and received by the Connection Target to open and establish explicit and I/O connections. The ForwardOpen Service request and associated response contains all of the connection parameters, including transport class, production trigger, timing information, electronic key and connection IDs.

Implicit Messaging
Implicit Messages are exchanged across I/O Connections with an associated Connection ID. The Connection ID defines the meaning of the data and establishes the regular/repeated transport rate and the transport class. No messaging protocol is contained within the message data as with Explicit Messaging. Implicit Messages can be point to point (unicast) or multicast and are used to transmit application specific I/O data. This term is used interchangeably with the term I/O Messaging. Implicit Messaging on EtherNet/IP uses UDP/IP frames on port 2222. They are typically Class 0 or 1 and of the type Exclusive Owner, Input Only and Listen Only.

Input Only Connection
This is one of three types of Implicit (I/O) Connections. It is a Class 0 or 1 Connection to an Input connection point (typically an assembly object). The scanner receives input data from the target device and produces a Heartbeat to the target device. There is no Output data.

I/O Adapter
An I/O Adapter receives implicit communications requests from an I/O Scanner then produces and consumes its I/O data, typically at the requested cyclic rate. An I/O Adapter can be a simple digital input device, or something more complex such as a modular pneumatic valve system.

I/O Client
Function that uses the I/O messaging services of another (I/O Server) device to perform a task. Initiates a request for an I/O message to the server module. The I/O Client is a Connection Originator of Implicit Message connections

I/O Messaging
Used interchangeably with the term Implicit Messaging.

I/O Scanner
An I/O scanner initiates implicit connections with I/O adapter devices, i.e., it is an I/O Client. A scanner is typically the most complex type of EtherNet/IP device, as it must deal with issues such as configuration of which connections to make, and how to configure the adapter device. Scanners also typically support initiating explicit messages, i.e., it is also an Explicit Message Client. A programmable controller is an example of an I/O scanner (used interchangeably with Scanner Class).

I/O Server
Function that provides I/O messaging services to another (I/O Client) device. Responds to a request from the I/O Client for an I/O connection. An I/O Server is the target of the implicit message connection request.

Listen Only Connection
This is one of three types of Implicit Connections. It is a Class 0 or 1 Connection to an Input connection point (typically an assembly object). The scanner receives input data from the target device and produces a Heartbeat to the target device. There is no Output data. A Listen Only Connection can only be attached to an existing Exclusive Owner or Input Only Connection. If this underlying connection closes, then the Listen Only connection will also be closed or timed out.

Master
EtherNet/IP does not use Master/Slave technology or terminology.

Message Client
Function that uses the Explicit messaging services of another (Message Server) device to perform a task. Initiates an Explicit Message request to the server device.

Message Server
Function that provides Explicit Messaging services to another (Message Client) device. Responds to an Explicit Message request from the Message Client.

Multicast
Multicast is the single transmission of an I/O data packet that may be consumed by multiple devices using multicast IP and Ethernet destination addresses. See Producer/Consumer Communications Model.

Producer/Consumer Communications Model
For I/O Connections, CIP supports object-oriented Producer/Consumer communication. Connection identifiers embedded into each message are used by devices to determine which messages they should "consume" from other devices that "produce" messages. This enables efficient use of network bandwidth by transmitting information only once. Less bandwidth equates to greater efficiency and overall speed. EtherNet/IP uses IP multicast and Ethernet multicast destination addressing to implement this capability.

Point to Point (Unicast)
Point to Point or Unicast is the transmission of data to a single device.

Requested Packet Interval (RPI)
EtherNet/IP devices typically produce or consume data based upon a Requested Packet Interval (RPI) value. Producer devices send data packets at a predetermined time interval based on the RPI, whereas consumer devices will listen for a packet of data at a given RPI.

Scanner Class
A Scanner Class product exchanges real-time I/O data with Adapter Class and Scanner Class products. This type of node can respond to connection requests and can also initiate connections to target devices (see I/O Scanner).

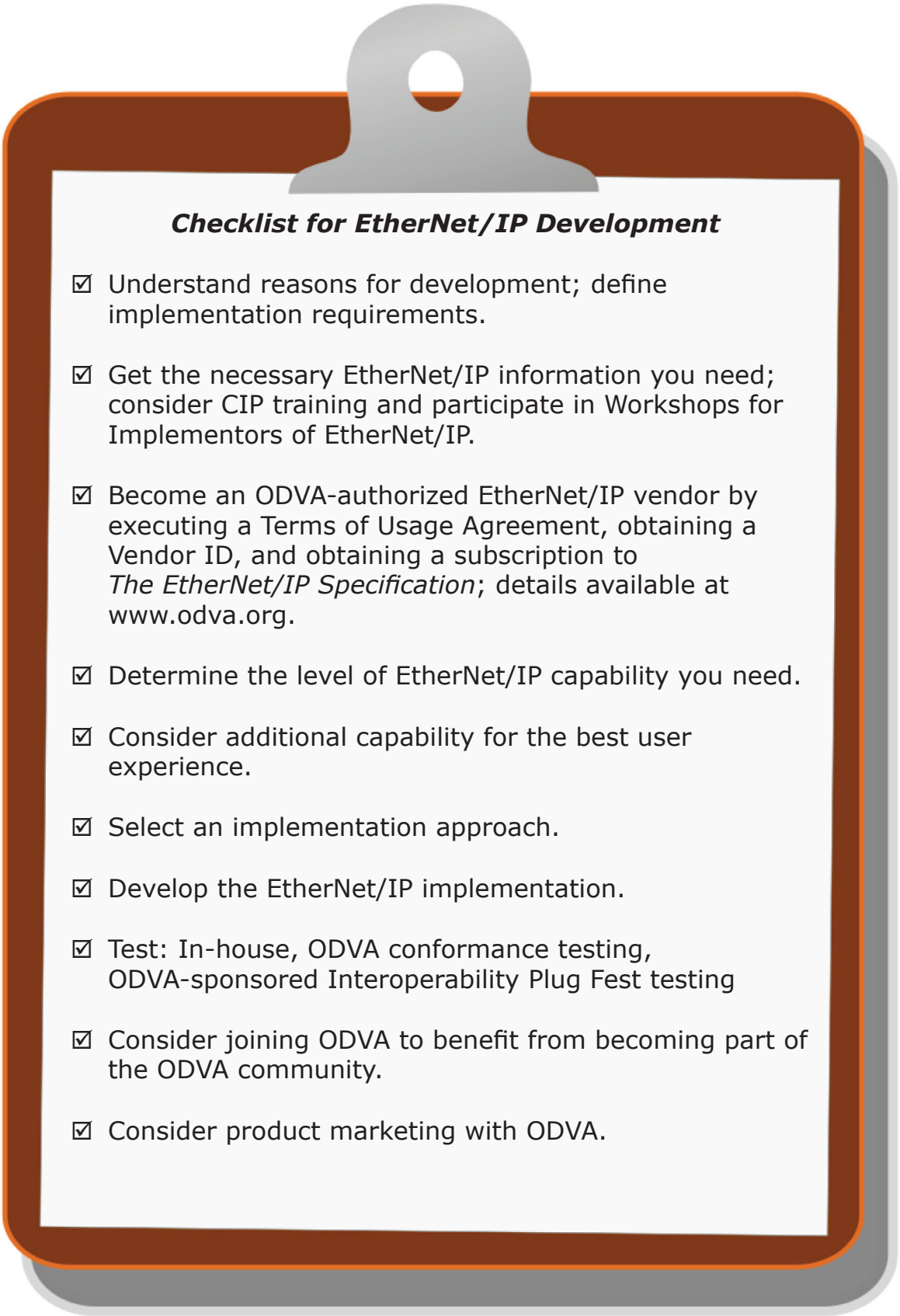
Slave
EtherNet/IP does not use Master/Slave technology or terminology.

Transport Classes
CIP defines several Transport Classes for messaging connections. Within EtherNet/IP, I/O data sent on Class 1 connections is pre-pended with a 16-bit sequence count, while data on Class 0 connections is not. Class 3 connections are used for Explicit Messaging Connections.

Unconnected Messaging
Provides a means for a node to send message requests without establishing a CIP connection prior to data transfer. More overhead is contained within each message and the message is not guaranteed destination node resources. Unconnected Messaging is used for non-periodic requests (e.g., network "Who" function). Applies to explicit messages only.

Unicast (Point to Point)
Unicast or Point to Point is a connection for the transmission of data to a single device.

EtherNet/IP Development Checklist



Checklist for EtherNet/IP Development

- ☑ Understand reasons for development; define implementation requirements.
- ☑ Get the necessary EtherNet/IP information you need; consider CIP training and participate in Workshops for Implementors of EtherNet/IP.
- ☑ Become an ODVA-authorized EtherNet/IP vendor by executing a Terms of Usage Agreement, obtaining a Vendor ID, and obtaining a subscription to *The EtherNet/IP Specification*; details available at www.odva.org.
- ☑ Determine the level of EtherNet/IP capability you need.
- ☑ Consider additional capability for the best user experience.
- ☑ Select an implementation approach.
- ☑ Develop the EtherNet/IP implementation.
- ☑ Test: In-house, ODVA conformance testing, ODVA-sponsored Interoperability Plug Fest testing
- ☑ Consider joining ODVA to benefit from becoming part of the ODVA community.
- ☑ Consider product marketing with ODVA.

Appendix A: Creating an EDS File

Why is an EDS file is important?

A well designed EDS will make device integration easy; a poor one will result in guesswork or can make device integration practically impossible in some cases. Here is some advice on how to create a well-designed EDS. The various parts of the EDS will be introduced in turn (as they appear in the EDS ASCII file, with brackets), describing their function and the information they contain.

- **[File] section.** This section is used for the administration of an EDS.

  If the keywords provided are not sufficient to document certain administrative details, then additional information can be added through the use of comments. It is strongly advised to use the URL keyword so that a user can check for the latest version of this EDS.

- **[Device] section.** The ID information contained in this section is used to match an EDS with a device found on the network.

  This section is a key element of any EDS. Identification is done by reading the first 4 attributes of the ID object and comparing this information with the equivalent information in the EDS (apart from the Minor Revision data component which is not used for this matching).

  Any devices that must be distinguished through their run-time options must therefore be covered by different EDSs and as a consequence, they must have different ID object attributes.

  Specifying an icon file name in the [Device] section will help assigning an icon automatically during the installation of the EDS. Having no icon is strongly discouraged, since icons are a good way to distinguish between device types/families in a graphical representation of a network, making identification easier for the user.

- **[Device Classification] section.** This section classifies the EDS/device for EtherNet/IP.

  This is a required for all EtherNet/IP devices. It must contain at least an entry for the EtherNet/IP network.

- **[Connection Manager] section.** This section specifies the CIP connections that may be made to the device.

  After the [File], [Device] and [Device Classification] sections, this is the most important section of an EtherNet/IP device that can be the target of any CIP connection. Only connections that are specified in this section can be used in an EDS-based configuration tool. While all kinds of Trigger & Transport types can be used for ConnectionN entries, it is common to only specify Class 0 and 1 connections within the EDS (Class 0 for safety connections, Class 1 for all other connections). No other transport classes can be used and interpreted by any EDS-based tool today.

  If multiple connections, with different options can be established into the device, then every connection needs a separate ConnectionN entry. In a few cases, ConnectionN entries can be "reused" for multiple connections, e.g. by using a parameter for the connection point information.

  The Transport Type used and the Connection Parameters chosen must yield a meaningful combination that matches the capabilities of the target device. If certain choices are mutually exclusive but supported by the device, then these choices must be covered by a set of individual ConnectionN entries. Tools like EZ-EDS can assist in avoiding some illegal combinations but not all of them. This tool also helps decoding the rather cryptic 32-bit values of the Trigger and Transport and Connection Parameter fields. Choosing meaningful names for the individual ConnectionN entries will help the user pick the right ones.

A path is required for all ConnectionN entries; otherwise, there would be no data components to connect to in the target device. It is highly recommended to support the full set of three application paths (configuration, consumed, produced), since this is one of the recommendations in ODVA publication 70, Recommended Functionality for EtherNet/IP Devices. Connections that are made to symbolic entities (tags) typically do not require a configuration path.

The Originator to Target (O→T) and Target to Originator (T→O) properties (RPI, Size and Format) can be used for some very meaningful information. When no RPI value is specified, a configuration tool may pick any value that can be supported for the given bus (network) and that could be a value beyond the device's capacity.

Using a fixed RPI value, does not make too much sense either, since that will be the only value that can be chosen. In most cases, it is best to use a ParamN entry inside the EDS to define the min/max/default values for RPI. For size and format, at least one of the two fields must be filled. If both fields are filled, the size field will take precedence; either only part of the data defined in the format field will be used or those bytes not covered by the format will be empty pad bytes. It is strongly recommended to define the format.

In the Configuration properties part, there is the capability to enter two configuration formats and two configuration sizes. This feature has been included to better handle modular devices. The first part of the resulting data attached to the Forward_Open request is meant to be consumed by the adapter while the second part of this data is to be forwarded to the individual modules according to their requirements. For non-modular devices, one part is sufficient.

- **[Assembly], [Params] and [ParamClass] section.** These sections should be filled as appropriate and as required by other parts of the EDS, e.g., ConnectionN entries.

  If parameter values are to be limited to a subset of the value range, as defined in the min/max fields of the ParamN entry, then enumeration can be a good way to do this.

  Configurable parameters for an EtherNet/IP device are expected to be packed into a Configuration Assembly. Individual parameters can be defined inside the EDS, but some tools on the market do not permit access to individual parameters inside the device even though access through Explicit Messaging (Get/Set_Attribute_Single or Get/Set_Attribute_All) will work.

- **[Capacity] section.** This section describes the communication capacity available within the device (and is therefore very useful to incorporate).

  Both the number of connections and the connection speeds (frames per second) can and should be described.

- **[Port] section.** This section provides port information and is really only required for devices that perform CIP routing.

  Although allowed, this section is unnecessary in devices supporting a single CIP port. In devices with a built-in switch, i.e., with multiple Ethernet ports, this section is still not required (or limited to one entry only) unless a CIP routing is performed from one port to another.

Appendix B: Extensions to CIP

CIP Safety, CIP Sync, and CIP Motion are extensions to CIP. Following is a brief overview of each.

What is CIP Safety?
CIP Safety is a high-integrity communication protocol built on top of standard CIP services. *The CIP Safety Specification* is Volume 5 of the CIP Networks Library. To develop a CIP Safety device using EtherNet/IP, you would also require Volumes 1 and 2. CIP Safety on DeviceNet and CIP Safety on EtherNet/IP have been certified for use in applications in systems needing to meet the requirements of IEC 61508 up to and including SIL3. IEC 61784-3 specifies Functional Safety Communication Profiles (FSCPs), i.e., extensions of fieldbusses for use in safety related applications. The CIP Safety protocol is included in IEC 61784-3, as FSCP 2/1.

What is required to develop a safety device on EtherNet/IP?
Two additional objects, a high integrity safety design, and certification by a notified body are required.

- Safety Validator Object: The safety validator object provides the high-integrity run-time communications behavior for safety devices. It executes the safety protocol behavior in the end devices and encapsulates and unencapsulates the safety protocol when it is being transmitted using standard CIP services. The Safety Validator simply ensures that data is transmitted with high-integrity (Figure 8).

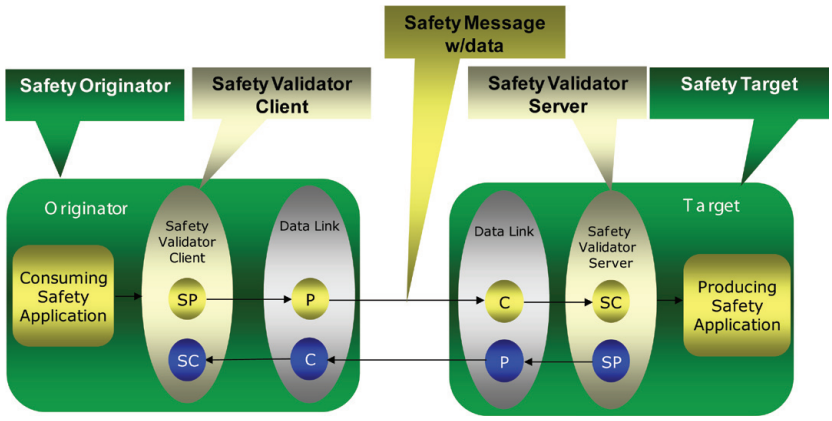


Figure 8
Safety validators in a safety device

- Safety Supervisor Object: The safety supervisor object provides the high-integrity configuration behavior for safety devices and assumes all mode control for the device. The Safety Supervisor ensures that the device is configured with high-integrity (Figure 13).

- High integrity safety design: High-integrity safety systems are typically systems with redundant hardware and well defined and controlled design processes.

- Certification: Typically safety designs are certified to ensure they meet the applicable standards by an independent agency or noted body. CIP Safety devices and typically certified to meet IEC 61508 and EN 954-1 by TÜV.

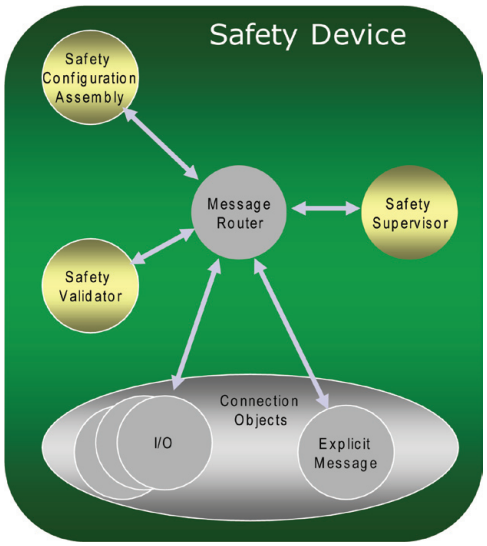


Figure 9
Configuring a safety device (Safety Supervisor)

What is the difference between integrity, reliability and availability?
It seems confusing but it is really pretty simple.

- Integrity is a measure of confidence, that is, what the probability is that something will happen as planned or go to a predefined state.

  For example a system that always returned the correct answer would have integrity of 1. A system that never returned the correct answer would have integrity of 0. Most real life systems are somewhere in-between. Safety systems have integrities very close to 1.

  If a system detects a failure, it will take a predefined action.

- Reliability is a measure of rate of failure. The mean-time between failures (MTBF) on a device may be 200,000 hours.

- Availability is a measure of what percent of the time is a system available for use to perform its desired function. It is measured in percent, that is, the system may be available 99.9% of the time.

Can I use the standard CIP services (e.g., seamless routing, multicast, etc.) with safety?
Yes. Since CIP Safety is built on standard CIP services, common CIP services, such as seamless routing and multicast, can be used.

Certification by Third-Party Competent Bodies
Development of CIP Safety nodes on EtherNet/IP shall be done according to the IEC 61508 requirements for SIL3 type software, including project management, verification and validation, and design and products shall be certified to be in compliance with IEC 61508 by an ODVA-approved competent body. In the case of CIP Safety, ODVA has approved the TÜV Rheinland Group (TÜV), headquartered in Cologne, Germany. Certification by TÜV is required by ODVA and TÜV in addition to certification by ODVA. The developer should consult TÜV whilst coordinating with ODVA.

Certification by ODVA
CIP Safety nodes on EtherNet/IP shall be certified by ODVA to be in compliance with *The EtherNet/IP Specification* and *The CIP Safety Specification*. Certification by ODVA is required by TÜV. The certification by ODVA covers only the network behavior of the product and its compliance with the Specifications. The developer should consult ODVA whilst coordinating with TÜV.

What does a CIP Safety toolkit implementation look like?
Although, there are many methods to achieve a SIL3 rating, an example toolkit architecture would specifically supply the following:

- Dual channel architecture (1oo2)
- Integrity of stack's binary image
- Integrity of non-volatile safety parameters storage
- Read/write interface between safe and non-safe environments (including access synchronization and data tearing prevention)
- CPU and memory (variables and code) integrity measures
- Program flow monitoring (Watchdog)
- Memory cross-checking
- Discrepancies are detected by either the safety application or the operating system and acted upon

This example toolkit architecture assumes that vendor's platform has been designed (in both hardware and software) to be compliant with requirements for SIL3 systems according to IEC 61508.

Example Toolkit: High Integrity Components
A wrapper around a generic CIP Safety stack (shown as the Platform Adaptation Layer in Figure 10 and Figure 11) provides translation facilities allowing for interfacing of generic toolkit code with functions provided by vendor's platform (including interfacing to safety related functions).

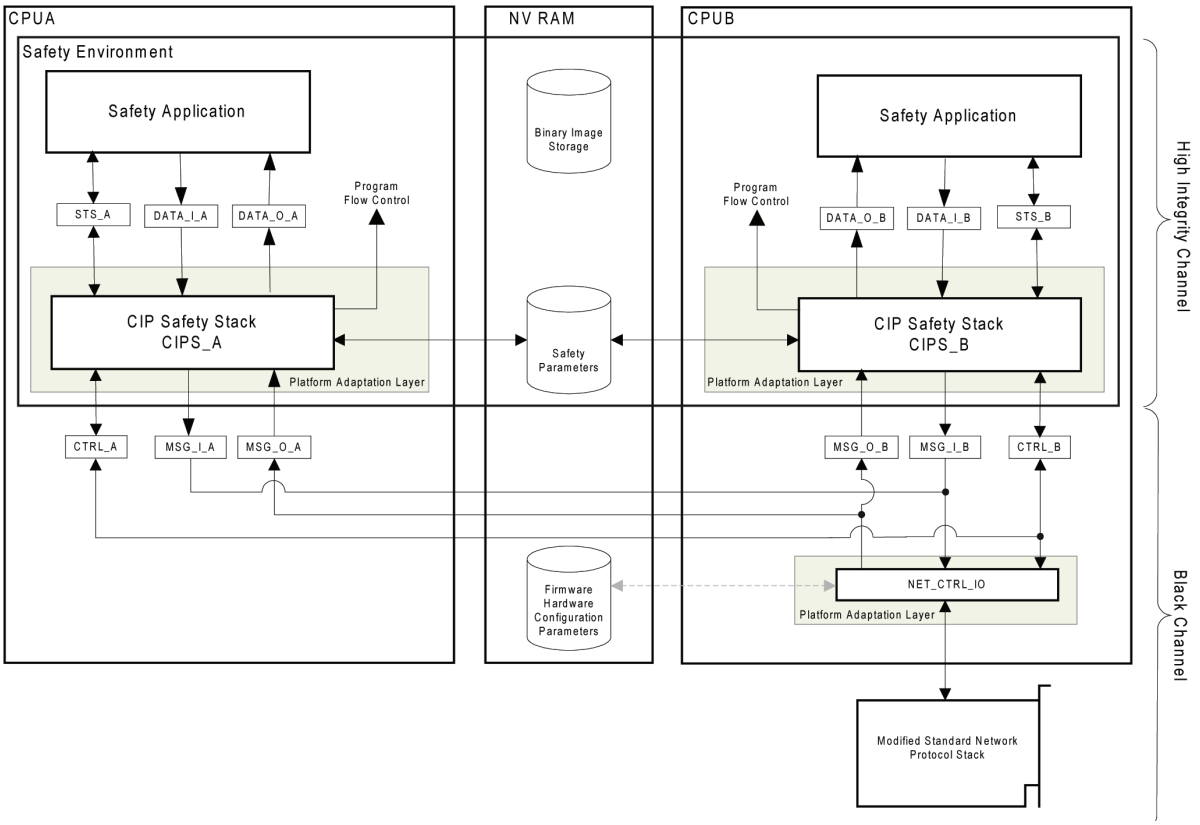


Figure 10
Example CIP Safety hardware architecture

The CIP Safety Stack comprises two functionally identical images CIPS_A and CIPS_B, developed according to a diverse programming approach. Message Validation and Production is initiated by the platform periodically.

Example Toolkit: Black Channel Components

The Network Control Task is responsible for communication with standard CIP protocol stack which could be located either on an external hardware device (DeviceNet Interface Card) or integrated as a part of the system (Software EtherNet/IP Stack). This task will periodically poll for events and data on the network and propagate them to the CIP Safety stack. At the same time this task will poll buffers for events and data generated by CIPS_n and if present propagate them to the standard stacks for network transmission. Both MSG_I_A and MSG_I_B must contain exact copies of a message for it to be sent out on the network.

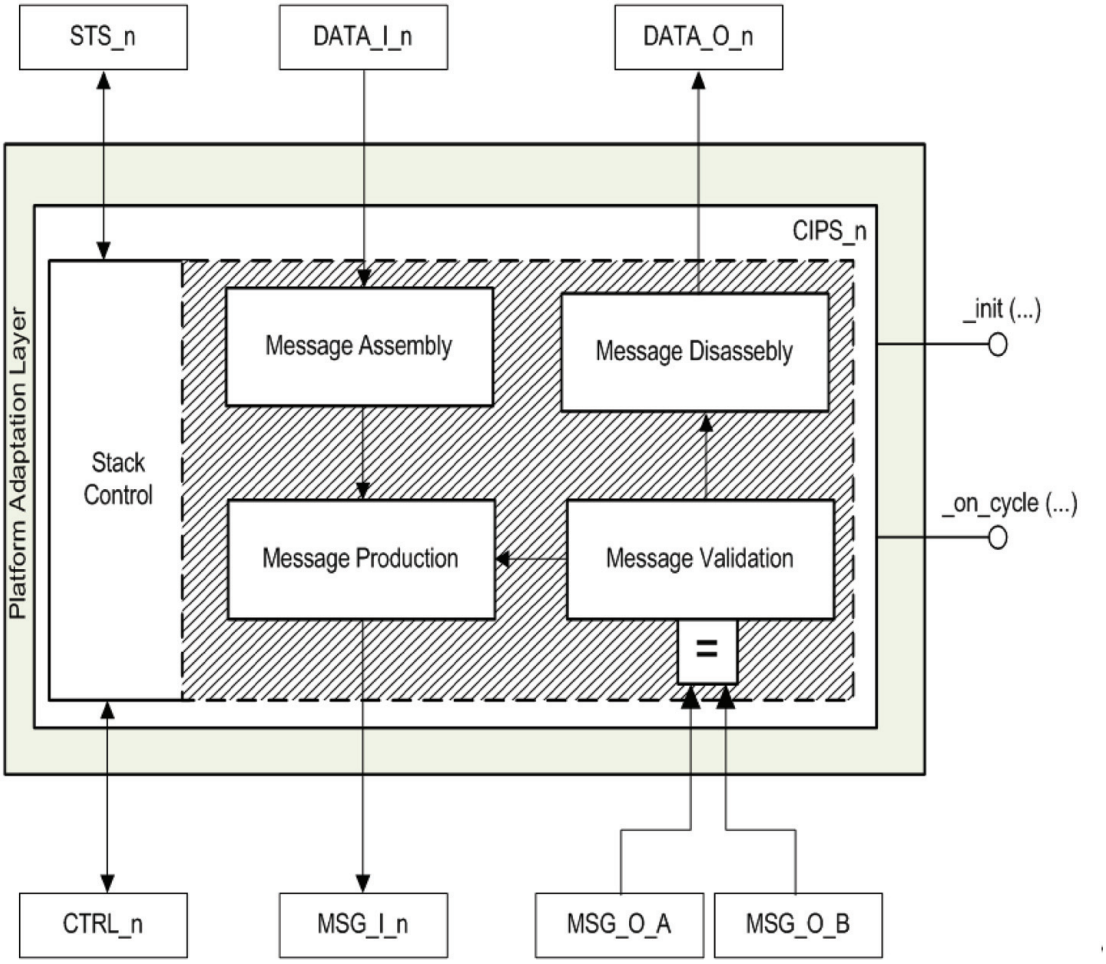


Figure 11
Example CIP Safety stack implementation

ODVA Resources for CIP Safety
The following documents are available from www.odva.org under the CIP Technology Library unless otherwise indicated. If an ODVA publication number (PUB) is assigned, it is indicated after the document's title.

To visit the CIP Technology Library, go to www.odva.org, and from the home page, click *'ODVA Technologies' → 'CIP' → 'CIP Technology Library'*

- ***The CIP Safety Specification***
  The specification contains all of the normative information for the CIP Safety protocol.

- ***CIP Safety White Paper***
  A short technical introduction to CIP Safety.

- ***The Common Industrial Protocol and the Family of CIP Networks*** *(ODVA PUB00123)*
  A detailed technical description of the CIP protocol and networks.

- ***DeviceNet Safety White Paper*** *(ODVA PUB00110)*
  A short technical introduction to CIP Safety on DeviceNet.

- ***Safety Standards***
  Overview presentation of international safety standards.

- ***EN954 Safety of Machinery***
  Overview presentation of EN954 machine safety standard.

- ***Safety Standards Evolution and Impact***
  Presentation on the evolution of international safety standards.

In addition, visit:

- **TÜV Rhineland**: www.tuvasi.com

- **IEC (International Electrotechnical Commission)**: http://www.iec.ch/

What is CIP Sync?

CIP Sync defines time synchronization services for CIP. These time services provide a distributed time reference for the packet time stamping used in the Time Synchronized Distributed Control scheme.

CIP Sync provides the increased control coordination needed for demanding events sequencing, distributed motion control and other highly distributed applications, where absolute time-synchronization of devices is vital.

CIP Sync forms the heart of EtherNet/IP's "Time Synchronized Distributed Control" for motion applications. In this approach, time-stamped data packets relax the strict requirement for less than 1µs jitter for cyclic data delivery. CIP Sync's time synchronization services provide a distributed time reference for the packet time stamping used in Time Synchronized Distributed Control. They also allow synchronization of services across distributed nodes.

CIP Sync also offers a method for synchronizing multiple clocks in a system. It is flexible enough to manage synchronization for applications that require time stamping, while, in the same system, servicing devices that depend on synchronized frequency.

CIP Sync & IEEE-1588™

CIP Sync is fully compliant with the IEEE-1588™ *Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems*, often referred to as the *Precision Time Protocol* (PTP). With this approach the CSMA/CD data link layer does not have to be replaced with a proprietary driver or ASIC, allowing full IEEE 802.3 compliance, while providing a robust solution with the performance necessary for closed loop operation of high performance digital drives (Figure 12).

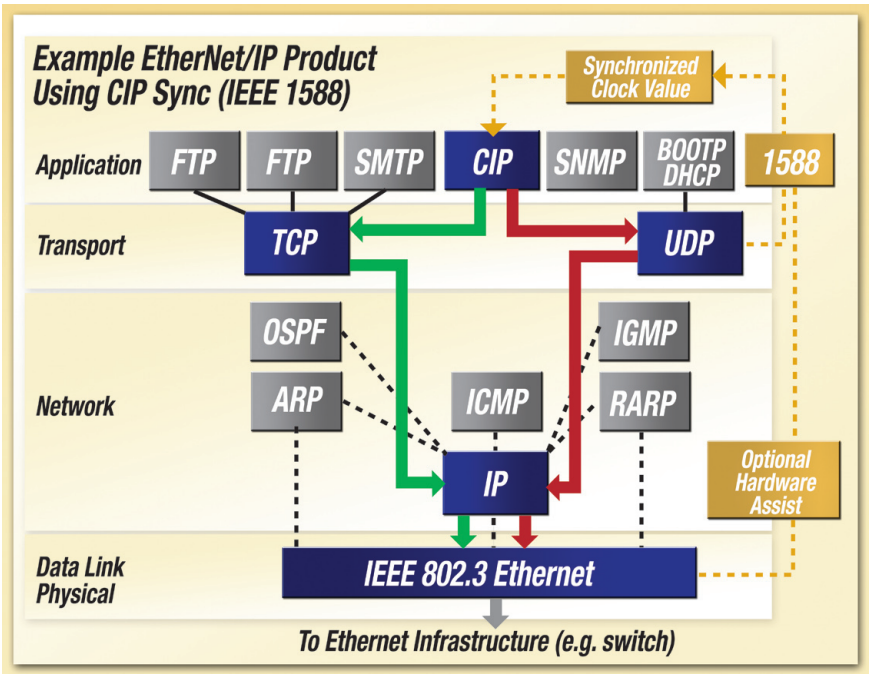


Figure 12
CIP Sync is fully compliant with the IEEE-1588™ Standard

What is CIP Motion?
CIP Motion defines extensions to standard CIP that are needed to realize high-performance motion control over EtherNet/IP. These extensions currently consist of a Motion Drive Device Profile and a Motion Axis Object specification.

What is required to develop a CIP Motion device on EtherNet/IP?
Implementation of a CIP Motion device profile is needed. Currently, the CIP Motion Drive Device Profile is defined and required to implement a CIP Motion compliant drive device. The CIP Motion profile defines extensions focused on drive control, including:

- Torque, velocity, or position control of servo and variable speed drives,
- Drive configuration, status, and diagnostic attributes and services
- Unicast control-to-drive communications
- Multicast peer-to-peer communications allow position and velocity synchronization in drives controlled by multiple distributed controllers
- Centralized and distributed motion support
- Common configuration, status and diagnostic services and common application instruction support for variable speed and servo drives makes those drives interchangeable at the application level

This device profile must include one or more instances of a key new CIP object component, the Motion Axis Object. In addition, high performance motion control requires accurate time synchronization between the motion controller and the drive device. This is achieved via the previously described CIP Sync protocol and its associated Time Sync Object.

Appendix C: Traffic Flow Diagrams

The basic sequence of messages for each fundamental type of EtherNet/IP messages are shown here. These are provided as supplemental material to expand some of the concepts discussed earlier.

Example 1
This is a sequence showing the steps needed to send an unconnected explicit message. This assumes that the two devices have not yet spoken so the TCP connection and Encapsulation Session have to be established. Normally, this is used for more than one explicit request, hence the indication of message iteration in the middle of the diagram.

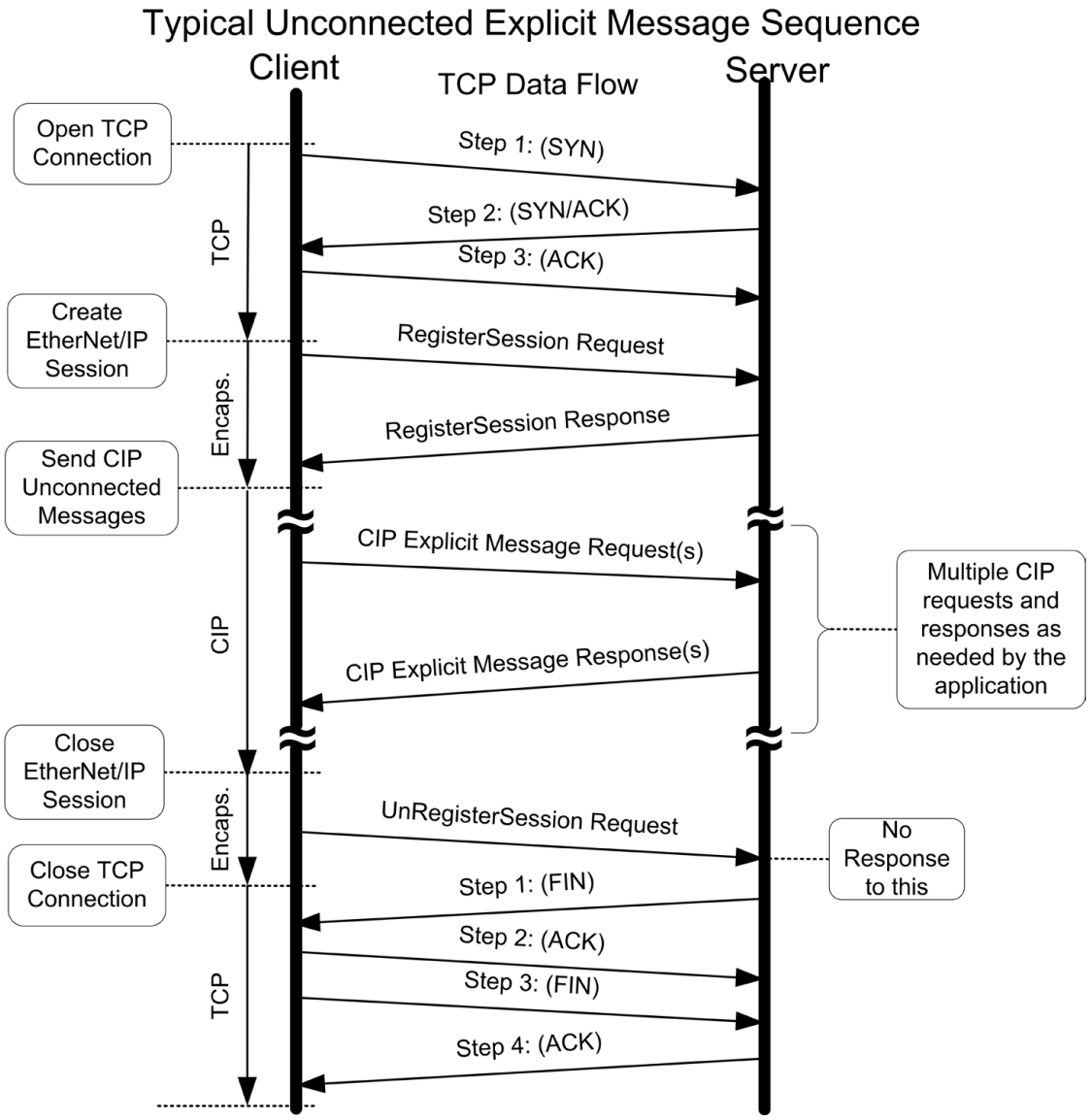


Figure 13
EtherNet/IP Sequence for Unconnected Explicit Messages

Example 2
This is a sequence showing the steps needed to open an explicit message connection. This assumes that the two devices have not yet spoken so the TCP connection and Encapsulation Session have to be established. An explicit message connection is normally used to send many explicit messages as indicated in the middle of the diagram. The number of requests and the duration of the connection is application dependent.

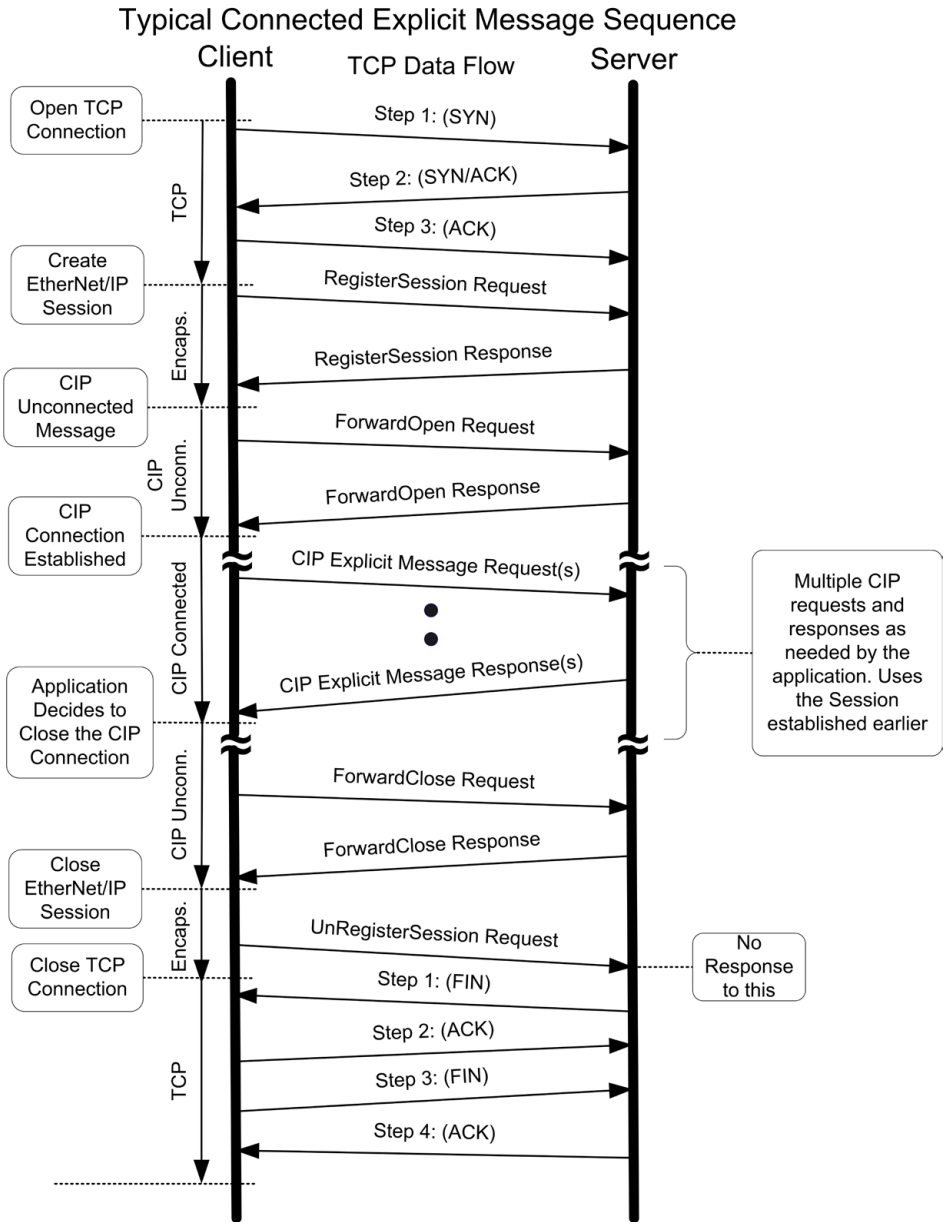


Figure 14
EtherNet/IP Sequence for Connected Explicit Messages

Example 3

This is a sequence showing the steps needed to open an I/O (Implicit) message connection. This assumes that the two devices have not yet spoken so the TCP connection and Encapsulation Session have to be established. An I/O connection is setup using an Explicit Message over TCP. Once established, the actual I/O connection runs over UDP, independent of the TCP connection (see UDP Data Flow in Figure 20). This I/O Connection will continue to exchange data indefinitely until it times out or it is intentionally discontinued with an Explicit Message.

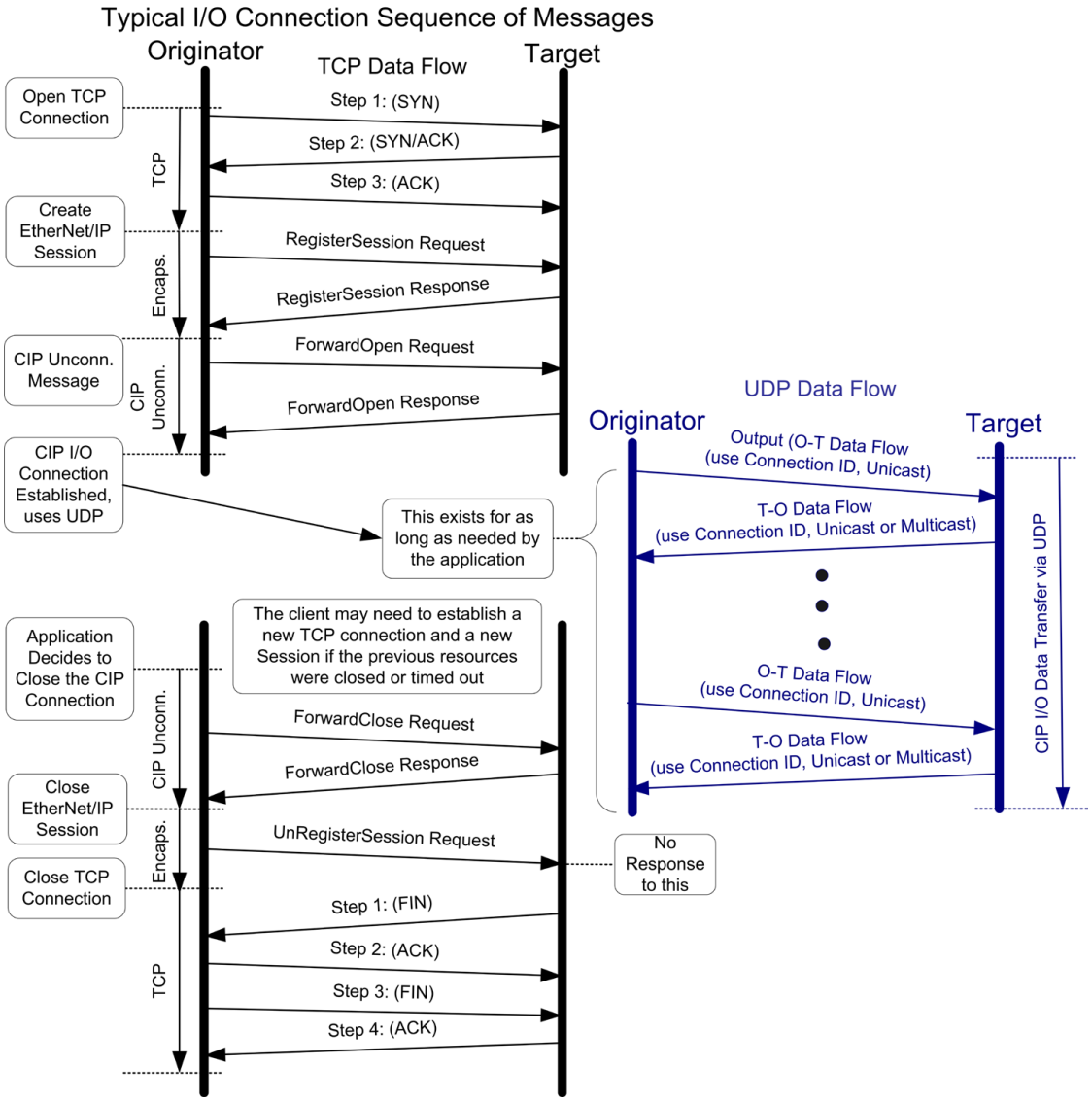


Figure 15
EtherNet/IP Sequence for I/O Data Transfer

Appendix D: Development Tools

Protocol analyzers available include:

- **Wireshark™**, with its built-in CIP decoder is a popular open source tool used to analyze EtherNet/IP traffic (Figure 9). Wireshark runs on Windows, Linux, UNIX, and other platforms. This tool was previously known as Ethereal.

- **NetDecoder™ (formerly FTS4Control)** is another protocol analysis tool that supports EtherNet/IP, CIP and other protocols.

Other development tools of note include:

- **EtherNet/IP Device Interoperability Test Tool (EDITT)** is a PC/Windows™-based software application that automates sections of the EtherNet/IP Interoperability Test Procedure, version 1.2. This test procedure is published by ODVA EtherNet/IP Implementors Workshop and performed during PlugFest interoperability testing. EDITT (Figure 10) is available from Pyramid Solutions (www.pyramid-solutions.com). EDITT provides EtherNet/IP I/O Server, I/O Client, Message Server and Message Client functionality. EDITT is capable of originating a variety of I/O connections based on the connection configuration set by the user. EDITT is compatible with Rockwell Software's RSNetWorx for EtherNet/IP for local or remote network configuration.

- **EtherNet/IP Scanner Simulation Tool (EIPScan)** is a PC/Windows application that simulates an EtherNet/IP Scanner Class device (connection client & server) to enable product engineers to test and debug EtherNet/IP connected products under development. EIPScan provides EtherNet/IP I/O Server, I/O Client, Message Server and Message Client functionality. EIPScan is capable of originating a variety of I/O connections based on the user set connection configuration. EIPScan is compatible with Rockwell Software's RSNetWorx for EtherNet/IP for local or remote network configuration. EIPScan is available from IXXAT (www.ixxat.de) and Pyramid Solutions (www.pyramid-solutions.com).

For more information, visit
www.odva.org