

Implantación del protocolo CIP sobre una plataforma embebida para EtherNet/IP



Diego Andrés Vásquez Escobar

Universidad del Cauca

**Facultad de Ingeniería Electrónica y Telecomunicaciones
Departamento de Electrónica, Instrumentación y Control
Ingeniería en Automática Industrial
Popayán, Febrero de 2013**

Implantación del protocolo CIP sobre una plataforma embebida para EtherNet/IP



Diego Andrés Vásquez Escobar

**Monografía presentada como requisito parcial para optar por el título de Ingeniería
en Automática Industrial**

Director: Ing. Vladimir Trujillo Arias

Universidad del Cauca

**Facultad de Ingeniería Electrónica y Telecomunicaciones
Departamento de Electrónica, Instrumentación y Control
Ingeniería en Automática Industrial
Popayán, Febrero de 2013**

Nota de aceptación:

Firma del presidente del jurado

Firma del jurado

Firma del jurado

Popayán, Enero de 2013

AGRADECIMIENTOS

A mi padre, que a la vida le doy infinita gracias por tenerlo. Su increíble generosidad, humildad, ternura e inmensa sabiduría, me enseña y me motiva cada día a seguir sus maravillosas cualidades y sobre todo seguir superándome académicamente, socialmente y espiritualmente.

A mi madre, por atenderme y preocuparse cada día en mis necesidades, haciendo siempre el mayor esfuerzo porque mis metas salgan adelante. También le agradezco por enseñarme a tener carácter frente a las circunstancias adversas que suceden en el largo camino de la vida.

A mi hermana, por ayudarme y aconsejarme incondicionalmente en el momento que yo lo necesito.

A mi sobrina preciosa, que me alegra y me motiva todos los días. Estaré a su lado por siempre para enseñarle y apoyarla en todo lo que la vida nos exige.

Y a mis familiares y amigos, que en etapas críticas de la vida me han dado su mano para poder seguir adelante.

CONTENIDO

	Pág.
1. LA RED INDUSTRIAL ETHERNET/IP.....	1
1.1 ETHERNET COMO UNA ARQUITECTURA INDUSTRIAL.....	1
1.2 DETERMINISMO CON ETHERNET INDUSTRIAL.....	1
1.3 TECNOLOGÍA ETHERNET/IP.....	3
1.4 LAS VENTAJAS DE CIP CON ETHERNET/IP.....	4
1.5 LA ESPECIFICACIÓN PARA LA RED ETHERNET/IP.....	6
2. DESCRIPCIÓN DEL PROTOCOLO INDUSTRIAL COMÚN CIP.....	6
2.1 TÉRMINOS GENERALES UTILIZADOS POR CIP.....	6
2.1.1 MODELO FUENTE/DESTINO.....	6
2.1.2 MODELO PRODUCTOR/CONSUMIDOR.....	7
2.1.3 MODELO PRODUCTOR/CONSUMIDOR PARA CIP.....	7
2.1.4 MENSAJE EXPLÍCITO.....	7
2.1.5 MENSAJE I/O O IMPLÍCITO.....	7
2.1.6 MODELO DE OBJETOS.....	7
2.1.7 PUNTO A PUNTO.....	7
2.1.8 MULTICAST.....	8
2.2 MODELADO DE OBJETOS.....	8
2.2.1 NODE ADDRESS O (NODE ID).....	9
2.2.2 CLASS ID.....	9
2.2.3 INSTANCE ID.....	10
2.2.4 ATTRIBUTE ID.....	10
2.2.5 SERVICE CODE.....	10
2.2.6 RANGO DE DIRECCIONES.....	10

2.3 MODELO DE OBJETOS CIP.....	12
2.3.1 CODIGO DE CLASE.....	12
2.3.2 ATRIBUTOS.....	13
2.3.2.1 ATRIBUTOS DE CLASE.....	13
2.3.2.2 ATRIBUTOS DE INSTANCIA.....	15
2.3.2.3 SERVICIOS.....	16
2.4 LIBRERÍA DE OBJETOS DE LAS REDES CIP...	18
2.4.1 CONJUNTO MÍNIMO DE OBJETOS REQUERIDOS EN UN DISPOSITIVO.....	20
2.5 PERFILES DE DISPOSITIVO.....	21
2.5.1 PERFIL DE DISPOSITIVO PARA ETHERNET/IP.....	23
2.5.1.1 DESCRIPCIÓN DE LOS ATRIBUTOS DEL OBJETO IDENTITY...	23
2.5.1.1.1 VENDOR ID.....	24
2.5.1.1.2 DEVICE TYPE.....	24
2.5.1.1.3 PRODUCT CODE.....	24
2.5.1.1.4 REVISION.....	24
2.5.1.1.5 STATUS.....	25
2.5.1.1.6 SERIAL NUMBER.....	25
2.5.1.1.7 PRODUCT NAME.....	26
2.5.1.2 DESCRIPCIÓN DE LOS ATRIBUTOS DEL OBJETO TCP/IP INTERFACE.....	26
2.5.1.2.1 STATUS.....	27
2.5.1.2.2 CONFIGURATION CAPABILITY.....	28
2.5.1.2.3 CONFIGURATION CONTROL.....	29
2.5.1.2.4 PHYSICAL LINK OBJECT.....	29

2.5.1.2.5 INTERFACE CONFIGURATION.....	29
2.5.1.2.6 HOST NAME.....	30
2.5.1.3 DESCRIPCIÓN DE LOS ATRIBUTOS DEL OBJETO ETHERNET LINK.....	30
2.5.1.3.1 INTERFACE FLAGS.....	31
2.5.1.4 DESCRIPCIÓN DEL OBJETO MESSAGE ROUTER.....	32
2.5.1.4.1 SOLICITUD DE SERVICIO.....	33
2.5.1.4.1 SOLICITUD DE RESPUESTA.....	33
2.5.1.5 DESCRIPCIÓN DEL OBJETO CONECTION MANAGER.....	33
2.6 ADAPTACIÓN DE LA RED ETHERNET/IP A CIP.....	34
2.6.1 USO GENERAL DE LA TRAMA ETHERNET.....	36
2.6.2 ENCAPSULAMIENTO DE MENSAJES.....	36
2.6.2.1 ESTRUCTURA DE ENCAPSULAMIENTO.....	36
2.6.2.1.1 LENGHT.....	37
2.6.2.1.2 SESSION.....	39
2.6.2.1.3 STATUS.....	39
2.6.2.1.4 SENDER CONTEXT.....	39
2.6.2.1.5 OPTIONS.....	39
2.6.2.2 DESCRIPCIÓN DE COMANDOS.....	39
2.6.2.2.1 NOP.....	39
2.6.2.2.2 LISTIDENTITY.....	39

2.6.2.2.3 REGISTERSESSION/UNREGISTERSESSION.....	40
2.6.2.2.4 SENDRRDATA/SENDUNIDATA.....	40
2.6.2.3 ADMINISTRACIÓN DE UNA SESIÓN.....	42
2.6.2.3.1 ESTABLECER UNA SESIÓN.....	44
2.6.2.3.2 MANTENER UNA SESIÓN.....	44
2.6.2.3.3. TERMINAR UNA SESIÓN.....	45
2.6.2.4 FORMATO DE PAQUETE COMÚN.....	45
2.6.2.4.1 ITEM DE DIRECCIONES.....	47
2.6.2.4.1.1 ITEM DE DIRECCIÓN NULA.....	47
2.6.2.4.1.2 ITEM DE DIRECCION CONECTADA.....	47
2.6.2.4.1.3 ITEM DE DIRECCIONES SECUENCIADAS.....	48
2.6.2.4.2 ITEMS DE DATOS.....	48
2.6.2.4.2.1 ITEM DE DATOS SIN CONEXIÓN.....	48
2.6.2.4.2.2 ITEM DE DATOS CON CONEXIÓN.....	48
2.7 ASIGNACIÓN DE LA MENSAJERÍA EXPLÍCITA PARA TCP/IP.....	49
2.7.1 PAQUETES CIP SOBRE TCP/IP.....	49
2.7.1.1 UNCONNECTED MESSAGES.....	49
2.7.1.2 TIPOS DE COMUNICACIONES ETHERNET/IP.....	51
2.7.1.2.1 MENSAJERÍA EXPLÍCITA.....	51
2.7.1.2.2 MENSAJERÍA IMPLÍCITA.....	52

2.7.2 INFORMACIÓN SOBRE EL COMPORTAMIENTO DEL TCP.....	52
2.8 DATA MANAGEMENT.....	53
2.8.1 USO DE LOS PATH PARA SEGMENTOS.....	54
2.8.1.1 FORMATO DEL PATH.....	54
2.8.1.2 TIPO DE SEGMENTO.....	55
2.8.1.2.1 LOGICAL SEGMENT.....	55
2.8.1.2.2 DATA TYPES.....	59
2.8.1.2.2.1 DATA TYPES ELEMENTALES.....	59
2.9 EDS ELECTRONIC DATA SHEETS.....	62
2.9.1 SECCIÓN DE UN EDS.....	64
2.9.2 REGLAS DE DISEÑO DE UN EDS.....	65
2.9.3 DESCRIPCIÓN DE LAS SECCIONES DE UN EDS.....	68
2.9.3.1 DESCRIPCIÓN DE LA SECCIÓN FILE.....	68
2.9.3.1.1 FILE DESCRIPTION TEXT.....	68
2.9.3.1.2 FILE CREATION DATE.....	69
2.9.3.1.3 FILE CREATION TIME.....	69
2.9.3.1.4 LAST MODIFICATION DATE.....	69
2.9.3.1.5 LAST MODIFICATION TIME.....	69
2.9.3.1.6 EDS REVISION.....	69
2.9.3.1.7 HOME URL.....	69
2.9.3.2 DESCRIPCIÓN DE LA SECCIÓN DEVICE.....	70

2.9.3.2.1 CATALOG NUMBER.....	71
2.9.3.2.2 ICON FILE NAME.....	71
2.9.3.3 DESCRIPCIÓN DE LA SECCIÓN DEVICE CLASIFICATION.....	71
2.9.3.4 DESCRIPCIÓN DE LA SECCIÓN PORT.....	72
2.10 CÓDIGOS DE ESTADOS GENERAL.....	73
3. LOS SISTEMAS EMBEBIDOS.....	78
3.1 DEFINICIÓN PARA SISTEMA EMBEBIDO.....	78
3.2 COMPONENTES DE UN SISTEMA EMBEBIDO.....	79
3.3 SISTEMAS EMBEBIDOS EN ETHERNET.....	80
4 DESARROLLO DEL PROTOCOLO CIP.....	81
4.1 ELEMENTOS ESCOGIDOS DEL PROTOCOLO CIP, PARA LA REALIZACIÓN DEL PROYECTO.....	83
4.1.1 VALORES ASIGNADOS DE LOS ATRIBUTOS DE INSTANCIAS DEL PC Y LA ND.....	84
4.1.1.1 ATRIBUTOS DE INSTANCIA DE LA ND+.....	84
4.1.1.1 ATRIBUTOS DE INSTANCIA DE LA PC.....	85
4.2 DIAGRAMAS UML DEL SOFTWARE DEL PROTOCOLO CIP.....	86
4.3 FORMATO SENDUNIDATA UTILIZADO PARA ESTE PROYECTO.....	87
4.4 VISTA DEL PROTOCOLO CIP IMPLANTADO.....	88
4.5 EDS DE LA PC Y LA NETDUINO PLUS.....	92
CONCLUSIONES.....	94
TRABAJOS FUTUROS.....	95

LISTA DE TABLAS

	Pág
TABLA 1. RANGOS PARA CLASE ID.....	10
TABLA 2 RANGOS PARA CÓDIGOS DE SERVICIO.....	11
TABLA 3.RANGOS PARA ATRIBUTOS ID.....	11
TABLA 4. DEFINICIONES PARA UN OBJETO.....	12
TABLA 5. CONTENIDO ATRIBUTOS DE CLASE.....	13
TABLA 6. ATRIBUTOS DE CLASE RESERVADOS PARA TODAS LASDEFINICIONES DE OBJETOS DE CLASE.....	14
TABLA 7. CONTENIDO ATRIBUTOS DE INSTANCIA.....	16
TABLA 8. CONTENIDO DE UN SERVICIO COMÚN.....	16
TABLA 9. NOMBRES Y CÓDIGOS DE SERVICIOS.....	17
TABLA 10. OBJETOS DEFINIDOS POR CIP PARA USO GENERAL.....	19
TABLA 11. OBJETOS ESPECÍFICOS DE APLICACIÓN.....	19
TABLA 12. OBJETOS ESPECÍFICOS DE RED CIP.....	20
TABLA 13. PERFILES DE DISPOSITIVOS CIP.....	22
TABLA 14. DIRECCIONES PARA PERFILES DE DISPOSITIVO.....	23
TABLA 15.ATRIBUTOS DE INSTANCIA DEL OBJETO IDENTITY.....	23
TABLA 16. DEFINICIÓN DE BITS PARA EL ATRIBUTO DE INSTANCIA STATUS, DEL OBJETO IDENTITY.....	25
TABLA 17.SERVICIOS SOPORTADOS POR EL OBJETO IDENTITY.....	26
TABLA 18.ATRIBUTOS DE INSTANCIA DEL OBJETO TCP/IINTERFACE.....	26
TABLA 19 ATRIBUTO STATUS.....	27

TABLA 21. ATRIBUTO CONFIGURATION CONTROL.....	28
TABLA 22. SERVICIOS SOPORTADOS POR EL OBJETO TCP/IP INTERFACE.....	28
TABLA 23. ATRIBUTOS DE INSTANCIA DEL OBJETO ETHERNET LINK.....	30
TABLA 24. SERVICIOS SOPORTADOS POR EL OBJETO ETHERNET LINK.....	30
TABLA 25. FORMATO DE SOLICITUD DEL MESSAGE ROUTER.....	31
TABLA 26. FORMATO DE RESPUESTA DEL MESSAGE ROUTER.....	32
TABLA 27. SERVICIOS ESPECÍFICOS DE OBJETO DEL CONNECTION MANAGER.....	34
TABLA 28. PAQUETE DE ENCAPSULACIÓN.....	37
TABLA 29. LISTA DE COMANDOS.....	37
TABLA 30. CÓDIGOS DEL ATRIBUTO STATUS DEL OBJETO IDENTITY.....	39
TABLA 31. SOLICITUD REGISTERSESSION.....	40
TABLA 32. REPLICA REGISTERSESSION.....	41
TABLA 33. COMANDO UNREGISTERSESSION.....	42
TABLA 34. SOLICITUD SENDRRDATA.....	43
TABLA 35. REPLICA SENDRRDATA.....	43
TABLA 36. COMANDO SENDUNIDATA.....	44
TABLA 37. FORMATO DE PAQUETE COMÚN.....	46
TABLA 38. FORMATO ÍTEM DE DATOS Y DIRECCION.....	46

TABLA 39. NÚMEROS DE LOS ÍTEM ID.....	46
TABLA 40. ÍTEM DE DIRECCIÓN NULA.....	47
TABLA 41. ÍTEM DE DIRECCIONES CONECTADAS.....	47
TABLA 42. ÍTEM DE DIRECCIÓN SECUENCIADA.....	48
TABLA 43. ÍTEM DE DATOS SIN CONEXION.....	48
TABLA 44. ÍTEM DE DATOS CON CONEXION.....	48
TABLA 45. SOLICITUD UCMM.....	49
TABLA 45. REPLICA UCMM.....	50
TABLA 47. COMUNICACIONES ETHERNET/IP.....	51
TABLA 48. FORMATO SEGMENTO ELECTRONIC KEY.....	57
TABLA 49. FORMATO KEY.....	57
TABLA 50. DATA TYPES ELEMENTALES.....	59
TABLA 51. CAMPOS DE LA SECCIÓN FILE.....	68
TABLA 52. CAMPOS DE LA SECCIÓN DEVICE.....	70
TABLA 53. CODIGOS DE ESTADOS GENERAL CIP.....	73
TABLA 54. COMANDO DE ENCAPSULACION UTILIZADO PARA ESTE PROYECTO.....	87

LISTA DE FIGURAS

	Pág
FIGURA 1.MODELO OSI DE LA RED ETHERNET/IP.....	3
FIGURA 2. NODO CIP SEGÚN EL MODELO DE OBJETOS.....	9
FIGURA 3.ESQUEMA DE DIRECCIONAMIENTO CIP.....	11
FIGURA 4. DISPOSITIVO TÍPICO REPRESENTADO SEGÚN EL MODELO DE OBJETOS CIP.....	21
FIGURA 5. PASOS PARA EL ESTABLECIMIENTO DE UNA CONEXIÓN EXPLICITA (SIN CONEXIÓN).....	35
FIGURA 6. RELACIÓN ENTRE CIP Y LA TRAMA ETHERNET.....	36
FIGURA 7. EJEMPLO DE PATH DE ATRIBUTO.....	54
FIGURA 8. CODIFICACIÓN DEL SEGMENTO BYTE TIPO/FORMATO.....	55
FIGURA 9. CODIFICACIÓN DEL LOGICAL SEGMENT.....	56
FIGURA 10. PACKET EPATH CON 8 BITS.....	58
FIGURA 11. PACKET EPATH CON 16 BITS.....	59
FIGURA 12. PADDED EPATH CON 16 BITS.....	59
FIGURA 13. ILUSTRACIÓN DEL USO DE UNA HOJA DATOS IMPRESA VS EDS.....	63
FIGURA 14. EJEMPLO DE EDS PARA UN DISPOSITIVO ETHERNET/IP.....	65
FIGURA 15. EJEMPLO DE LA SECCIÓN PORT.....	73
FIGURA 16. EJEMPLO DE UNA “DECLARACIÓN DE CONFORMIDAD DE LA ESPECIFICACIÓN ETHERNET/IP”.....	82
FIGURA 17. DIAGRAMA DE CASO DE USO.....	86

FIGURA 18. DIAGRAMA DE OBJETOS PARA LOS ATRIBUTOS DEL PC.....	86
FIGURA 19. DIAGRAMA DE OBJETOS PARA LOS ATRIBUTOS DE LA NETDUINO PLUS.....	87
FIGURA 20. DIRECCIONAMIENTO CIP PARA EL ATRIBUTO VENDOR ID DE LA ND+.....	88
FIGURA 21. UBICACIÓN DEL ATRIBUTO VENDOR ID DE LA ND+.....	89
FIGURA 22. UBICACIÓN DEL ATRIBUTO VENDOR ID EN MODO CONSOLA PARA LA ND+.....	89
FIGURA 23. UBICACIÓN DEL ATRIBUTO VENDOR ID EN MODO CONSOLA PARA LA ND+.....	90
FIGURA 24. DIRECCIONAMIENTO CIP PARA EL ATRIBUTO VENDOR ID PARA EL PC.....	90
FIGURA 25. ACTIVACIÓN DEL OBJETO IDENTITY DEL PC.....	91
FIGURA 26. FINALIZACIÓN DE LA CONEXIÓN.....	91
FIGURA 27. EDS NETDUINO PLUS.....	92
FIGURA 28. EDS PC.....	93

LISTA DE ANEXOS

Anexo A. Selección y descripción de la tarjeta embebida para el desarrollo del proyecto.

Anexo B. Experiencia en el manejo de sockets en C# y forma de aplicación en la Netduino Plus.

Anexo C. Experiencia en la implantación del protocolo CIP en el PC y la Netduino Plus.

Anexo D. Código fuente en C# del proyecto implantación del protocolo CIP sobre una plataforma embebida para EtherNet/IP.

1. LA RED INDUSTRIAL ETHERNET/IP

1.1 Ethernet como una arquitectura industrial

Ethernet es el protocolo por el cual se comunican ordenadores en un entorno de red local, es decir, es el sistema que normalmente se utiliza para comunicar ordenadores entre sí dentro de una industria. Este protocolo permite compartir la información y manejar completamente un ordenador o cualquier tipo de maquinaria desde otro. Además, podremos compartir el acceso a internet, de forma que con una sola conexión a internet puedan trabajar varios ordenadores. Todo ello ha abierto las puertas a algo fundamental, la disponibilidad de información remota y en tiempo real. Debidamente procesada, la información sobre el funcionamiento de una planta industrial puede estar disponible en cualquier momento y en cualquier sitio.

La introducción de Ethernet ha tenido un profundo impacto en la industria. Debido a sus capacidades de control de planta y oficina, aporta a los fabricantes una gran cantidad de ventajas que incluyen una integración más fácil entre los sistemas de planta y de administración, y la posibilidad de utilizar una sola infraestructura de red para funciones distintas.

Ethernet, según el protocolo de control de transmisión (TCP) y el protocolo de internet (IP), es una tecnología usada ampliamente para comunicaciones controlador a controlador y controlador a servidor y se usan cada vez más en el segmento de la automatización de entrada/salida (I/O). Una preocupación principal de los ingenieros que quieren usar internet en I/O es el determinismo y previsibilidad para el control industrial. Ethernet está basado en la teoría de probabilidades. Los datos son almacenados en paquetes y los paquetes son transmitidos por la red y sólo son reensamblados cuando llegan a su destino.

Si la red está muy ocupada cuando se intenta enviar un paquete, el sistema reintentará enviarlo hasta que encuentre un espacio. Surgen muchas veces problemas en automatización por no habernos planteado estas cuestiones: ¿El paquete estará en el punto I/O en el momento justo para que el dispositivo haga lo que está previsto, y en el momento exacto en que está previsto? [1]

1.2 Determinismo con Ethernet industrial

Los medios físicos de Ethernet – el cable y los conectores utilizados por los PCs en las oficinas, impresoras y demás dispositivos periféricos, trabajan con una gama de protocolos de comunicación tales como IP (Protocolo Internet), TCP

(Protocolo de Control de Transmisión) y muchos otros protocolos de envío de información por red. Estos tipos de protocolos van muy bien en el ambiente de oficina.

Además, permiten que los usuarios compartan archivos, accedan a impresoras, envíen e-mails, naveguen por Internet y realicen todo tipo de comunicación normal en un ambiente de oficina. Sin embargo, las necesidades a pie de fábrica son mucho más exigentes y demandan la adecuación a algunos requerimientos especiales. A pie de fábrica, los controladores tienen que acceder a datos en los mismos sistemas operativos, estaciones de trabajo y dispositivos I/O. En una situación normal, los programas dejan al usuario esperando mientras realizan su tarea. Pero en planta todo es distinto. Aquí el tiempo es crucial y ello requiere una comunicación en tiempo real. Por ejemplo, para un robot soldador o la operación de rellenar una botella en su tiempo justo requiere un ajuste de tiempo sumamente preciso, comparativamente a lo que se exige para acceder a un archivo en un servidor remoto o sencillamente hojear un Web por Internet.

Ethernet industrial es por su comportamiento una red no determinista.

Presenta un comportamiento inadecuado para aplicaciones de control sometidas a tráfico moderado y pesado. Cuando la carga de la red aumenta sobre el 50% se produce lo que se llama el efecto "trashing".

Es decir, a medida que aumenta la carga el rendimiento de la red disminuye y los retrasos máximos inducidos aumentan de forma aleatoria según aumenta la carga. Como consecuencia de este efecto, se presentan tasas elevadas de pérdidas de paquetes, inclusive llegando al extremo que el sistema puede fallar completamente. [2]

Por todo esto existen aún algunas barreras a la aceptación de Ethernet en el ambiente industrial, pero eso se debe a la falta de un nivel aceptable de software en las plantas y la falta de conocimiento acerca de la conectividad ofrecida por Ethernet en la automatización industrial.

En conclusión, Ethernet y TCP/IP tan solo garantiza que los dispositivos Ethernet coexistan en la red, por lo que no certifica que estos módulos se comuniquen de forma efectiva.

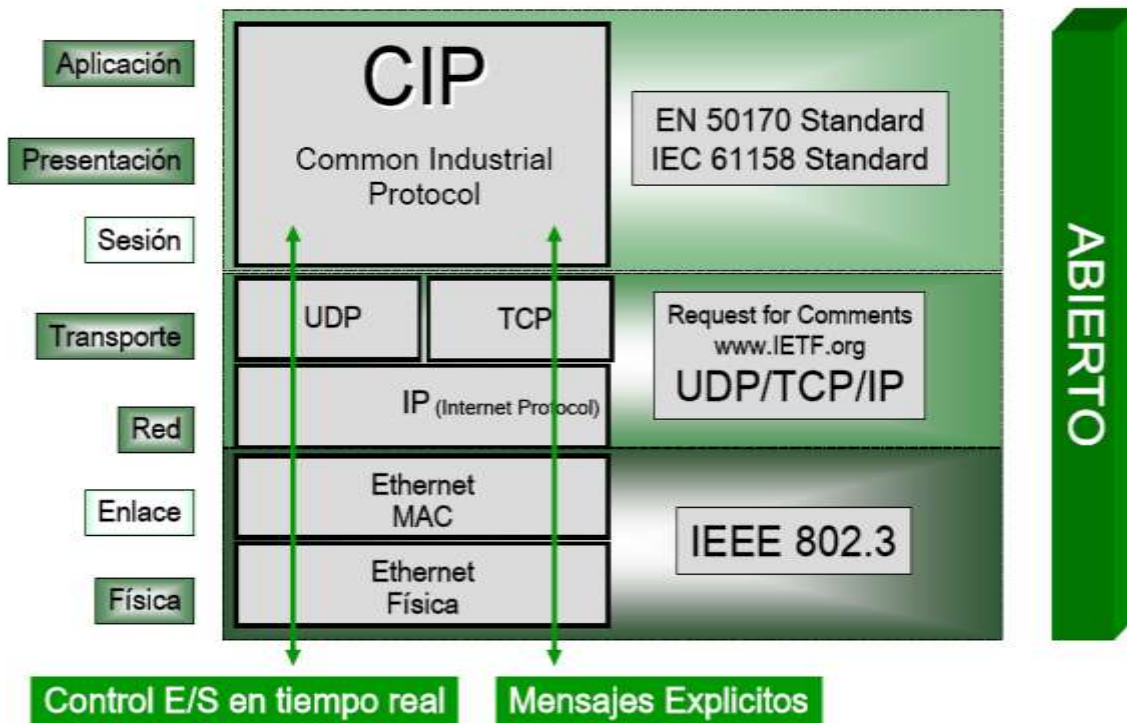
Los arquitectos de TCP/IP entendieron este problema y diseñaron nuevos protocolos para trabajar con múltiples capas de aplicación. Actualmente hay varias de las cuales podemos reconocer: FTP (para transferencia de archivos), STMP (email), HTTP (world wide web), OPC entre otras.

1.3 Tecnología Ethernet/IP

Ethernet/IP es un protocolo de red en niveles, apropiado al ambiente industrial. Es el producto acabado de cuatro organizaciones que aunaron esfuerzos en su desarrollo y divulgación para aplicaciones de automatización industrial: La Open DeviceNet Vendor Association (ODVA), la Industrial Open Ethernet Association (IOANA), la Control Net International (CI) y la Industrial Ethernet Association (IEA), que la lanzaron al mercado en marzo de 2000. Está basado en tecnología abierta, utilizando la misma capa de aplicación que utiliza las redes industriales DeviceNet [3], ControlNet [4] y la más reciente CompoNet [5], el Common Industrial Protocol (CIP).

Es importante entender la tecnología básica tras el nombre antes de entrar en detalles. Ethernet/IP se define mejor analizando su nombre. La porción "Ethernet" se refiere al estándar comercial IEEE 802.3 (no a una versión modificada o propietaria) y al conjunto de protocolos TCP/IP y UDP/IP. Por otro lado, "IP" son las siglas de Industrial Protocol y es lo que distingue a esta red. A diferencia de otras opciones industriales de Ethernet, Ethernet/IP utiliza un protocolo abierto como capa de aplicación (CIP). Por lo tanto podemos decir que la red Ethernet/IP es el protocolo de aplicación CIP, implementado en una red Ethernet TCP/IP. [6]

FIGURA 1: Modelo Osi de la red EtherNet/IP



Fuente propia

1.4 Las ventajas de CIP con Ethernet/IP:

Son muchas las ventajas del nivel del Protocolo de Control e Información (CIP) sobre Ethernet/IP. La oferta de un acceso consistente a aplicaciones físicas significa que se puede utilizar una sola herramienta para configurar dispositivos CIP en distintas redes desde un único punto de acceso sin la necesidad de software propietario. Todo nodo CIP es modelado como una “colección de objetos” (en el siguiente punto explicaremos esto más detalladamente); al clasificar todos los mecanismos como objetos o elementos, se reduce la necesidad de adiestramiento y los costos de puesta en marcha requeridos cuando se incorporan nuevos mecanismos al perímetro de la red. Ethernet/IP disminuye el tiempo de respuesta e incrementa la capacidad de transferencia de datos respecto al DeviceNet o al ControlNet. A través de un mismo medio de interconexión, Ethernet/IP conecta distintos mecanismos industriales con el control de planta y con la gestión central, mediante una interfaz consistente con las aplicaciones [7].

El protocolo CIP ofrece una amplia gama de servicios estándar para acceder a los datos y controlar los dispositivos conectados a la red a través de mensajes “implícitos” y “explícitos”. CIP utiliza mensajes implícitos para regular el intercambio de datos cíclicos en los que las estaciones implicadas están al tanto de los datos que se esperan: bloques de datos puros muy compactos con muy poca cabecera, datos típicos de E/S a través de conexiones de E/S. Todos los mensajes individuales que se envían una única vez utilizan el tipo explícito asociado, por ejemplo, todos los mensajes pregunta-respuesta entre el cliente y el servidor.

Como ya lo habíamos mencionado Ethernet/IP puede considerarse como la ampliación industrial de Ethernet TCP/IP, puesto que los mensajes CIP de la capa de aplicación se “empaquetan”, mediante encapsulación, en las tramas TCP/IP como datos de usuario. De este modo, una aplicación puede enviar sus datos a otra aplicación a través de Ethernet; en caso necesario, la aplicación genera automáticamente un mensaje CIP y lo encapsula para convertirlo en un paquete TCP/IP, algo similar a introducir una carta en un sobre. Este mensaje empaquetado se envía a través de Ethernet hasta el dispositivo de destino en el que, una vez recibido, el protocolo TCP/IP lo vuelve a enviar al protocolo de encapsulación para “desempaquetar” el mensaje original CIP (sacarlo del sobre) y volver a enviarlo, a través del protocolo CIP, hasta la aplicación receptora. En principio, es posible establecer este tipo de enlace de aplicación entre todos los usuarios del protocolo de aplicación CIP, incluso si proceden de distintos fabricantes o están ubicados en redes diferentes.

Esto significa que a través de TCP/IP, Ethernet/IP puede enviar “mensajes explícitos”, que es el nombre que reciben los mensajes a modo de telegramas que contienen información e instrucciones sobre el protocolo preciso para su posterior utilización en el campo de datos. El receptor debe interpretar los mensajes explícitos como indicaciones, ejecutarlas y generar una respuesta.

Este modo versátil de intercambio de datos se utiliza, por ejemplo, para la configuración, programación y diagnóstico de dispositivos con cantidades variables de datos.

No obstante, la comunicación en tiempo real impone requisitos algo distintos.

En ese caso, Ethernet/IP no utiliza el protocolo TCP sino UDP vía IP (Internet Protocol). Básicamente, este protocolo resulta más compacto, por lo que es compatible con los denominados mensajes “multidifusión” (recepción simultánea por varios usuarios) y puede ser utilizado por Ethernet/IP para el envío de los denominados “mensajes implícitos”. En este tipo de mensajes telegráficos, los campos de datos ya no incluyen la información de protocolo sino sólo datos de entrada y salida en tiempo real. La aplicación receptora ya conoce cómo debe interpretar estos datos, puesto que ya lo ha negociado durante la configuración de la conexión. Es decir, que los telegramas implícitos se envían a través de una conexión virtual existente entre los usuarios y se actualizan de manera constante y cíclica en cortos intervalos de tiempo con señales de E/S y datos recientes y actualizados. En este caso la cabecera es mínima, a fin de que los mensajes se procesen muy rápidamente y con prioridad (ese es justamente el requisito de las tareas de control en las que el tiempo es crítico).

En consecuencia, Ethernet/IP combina el protocolo TCP/IP y los telegramas de datos UDP/IP para el transporte de paquetes de mensajes explícitos e implícitos, lo que significa que, en este caso, tanto los datos de entrada y salida en tiempo real para las tareas de control prioritarias (UDP) como los datos de información (TCP) de una red pueden utilizarse en paralelo.

Por consiguiente, Ethernet/IP resulta ideal para las tareas de control de E/S, configuración y diagnóstico, y para la adquisición de datos en entornos industriales, especialmente si se tiene en cuenta la interoperabilidad e intercambiabilidad de un protocolo internacional de automatización.

1.5 La especificación para la red Ethernet/IP

Las especificaciones para cada red que trabaje con CIP, son entregadas por la ODVA y ControlNet International (ODVA/CI) en forma de Volúmenes. Así, cada red, se compone de dos volúmenes: un volumen común (capas de aplicación), y un volumen específico (capas inferiores).

Es así que la red Ethernet/IP está compuesta por dos especificaciones:

- Volumen 1: Protocolo Industrial Común.
- Volumen 2: Adaptación EtherNet/IP a CIP.

Cabe mencionar que el Volumen 1 está estandarizado internacionalmente a través del Comité Electrotécnico Internacional, o IEC (por sus siglas en inglés), que corresponde al IEC-61158 type 2, del año 2000, y en ella se detallan los aspectos comunes de CIP que se aplican a todas las adaptaciones de la red.

En el Volumen 2, que es la adaptación EtherNet/IP de CIP, se describe cómo CIP es adaptado a las capas de transporte Ethernet TCP/IP y de UDP/IP.

También contiene cualquier extensión necesaria para el Volumen 1 que sea necesaria para EtherNet/IP.

2. DESCRIPCION DEL PROTOCOLO INDUSTRIAL COMUN CIP

***Nota:** Será descrito según la edición 3.3 de Noviembre de 2007, para el Volumen 1, y la edición 1.4 de Noviembre de 2007 para el Volumen 2; que son las versiones más recientes que se consiguen en internet.*

2.1 Términos generales utilizados por CIP:

2.1.1 Modelo Fuente/Destino: Es un modelo de comunicación que establece la forma en que son compartidos los mensajes. En este modelo, el dispositivo **fuentes** envía un mensaje a un solo dispositivo destinatario específico. Por otra parte, el dispositivo destino es el que recibe el mensaje procedente desde el dispositivo fuente. Por esta razón, se incluye en la cabecera de un mensaje, la dirección de red del dispositivo fuente y la dirección de red del dispositivo destino. Este tipo de comunicación es punto a punto.

2.1.2 Modelo Productor/Consumidor: Es un modelo de comunicación que establece la forma en que son compartidos los mensajes. En este modelo, el dispositivo **productor** coloca un mensaje sobre la red para el consumo por uno o varios consumidores. Generalmente, el mensaje producido no se dirige a un consumidor específico. Por otra parte, el **consumidor** es un dispositivo que recoge (*consume*) un mensaje puesto en la red por un dispositivo productor.

El consumidor determina qué mensaje consumir por medio de un identificador en la cabecera del mensaje. Este modelo de comunicación es inherentemente *multicast* y puede soportar además comunicación punto a punto (por lo tanto, puede decirse que incorpora al modelo fuente/destino).

2.1.3 Modelo productor/consumidor para CIP: El protocolo CIP utiliza el modelo de comunicación productor/consumidor, en contraste con el modelo tradicional de fuente/destino.

Aprovecha su naturaleza *multicast*. Los nodos sobre la red determinan si se deben consumir los datos de un mensaje basándose en un valor de identificación (*identifier*) de la conexión, el cual está incluido en el paquete.

2.1.4 Mensaje explícito: Los mensajes explícitos contienen la información de dirección y de servicio que lleva al dispositivo receptor a realizar cierto servicio (*una acción*) en una parte específica de un dispositivo (por ejemplo, un atributo).

2.1.5 Mensaje I/O o implícito: Los mensajes I/O o implícitos, no llevan la información de dirección y/o de servicio; el nodo consumidor sabe qué hacer con los datos basados en la Conexión ID que fue asignada cuando la conexión fue establecida. Los mensajes implícitos son nombrados así porque el significado de los datos está implícito por la conexión ID.

2.1.6 Modelo de Objetos: El protocolo CIP utiliza el modelo de objetos para la implementación de las capas de aplicación de las distintas redes de campo. El modelo de objetos se basa en representaciones abstractas de las características de los dispositivos conectados a determinada red. Dado que estos objetos se ubican en las capas de aplicación de acuerdo al modelo ISO/OSI, un objeto es creado a través del uso de algún lenguaje de programación apropiado (por ejemplo, C++, C#) siguiendo las pautas de estructura y sintaxis entregadas por las especificaciones de CIP.

2.1.7 Punto a punto: Es el envío de información entre un dispositivo emisor (por ejemplo, un Maestro que envía un mensaje de solicitud de información) hacia un

solo dispositivo receptor (por ejemplo, un esclavo que recibe el mensaje y luego reenvía la información solicitada).

2.1.8 Multicast: Es el envío de información desde un dispositivo emisor (por ejemplo, un cliente que envía un mensaje de solicitud de información) hacia varios dispositivos receptores determinados (por ejemplo, varios dispositivos servidores).

2.2 Modelado de Objetos:

El protocolo CIP utiliza un modelo de objetos abstractos para describir:

- El conjunto de servicios de comunicación disponibles.
- El comportamiento visible externamente de un nodo CIP.
- Un medio común para el acceso e intercambio de información entre productos CIP.

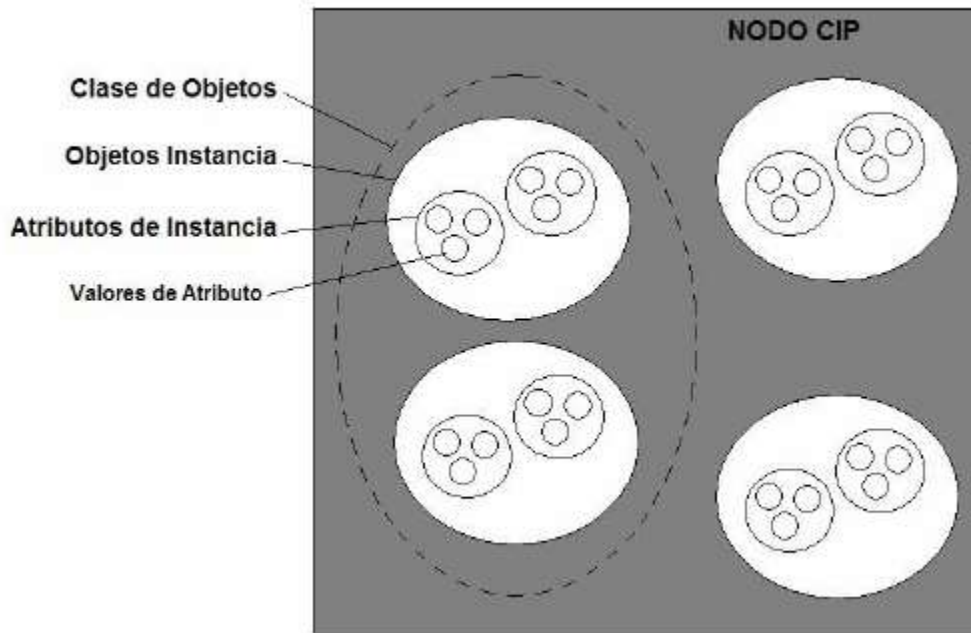
Todo nodo CIP es modelado como una *colección de objetos*. Un objeto provee una representación abstracta de un componente particular dentro de un producto. Cualquier cosa no descrita en forma de objeto no es visible a través de CIP.

Los objetos CIP están estructurados en los siguientes elementos:

- Clases**
- Instancias**
- Atributos**

Una **clase** es un conjunto de objetos donde todos representan el mismo tipo de componente de sistema. Un objeto **instancia** es la representación real de un objeto particular dentro de una clase. A su vez, cada instancia de una clase posee los mismos **atributos**, y que a su vez poseen su propio conjunto particular de valores.

FIGURA 2: Nodo CIP según el modelo de objetos



Fuente: Tomado de [8] 13 de Septiembre de 2012

Podemos observar que múltiples objetos instancia dentro de una clase particular pueden residir dentro de un nodo CIP.

Los objetos y sus componentes son direccionados a través de un esquema uniforme compuesto de los siguientes elementos:

- **Node Address** (Dirección de Nodo)
- **Class Identifier o Class ID** (Identificador de Clase)
- **Instance Identifier o Instance ID** (Identificador de Instancia)
- **Attribute Identifier o Attribute ID** (Identificador de Atributo)
- **Service Code** (Código de Servicio)

2.2.1 Node Address (o Node ID): Dirección de Nodo. Es un valor entero de identificación asignado a cada nodo sobre una red CIP. En EtherNet/IP, la dirección de nodo es la dirección IP.

2.2.2 Class ID: Identificador de clase. Es un valor entero de identificación asignado a cada clase de objetos accesible desde la red.

2.2.3 Instance ID: Identificador de Instancia. Es un valor entero de identificación asignada a un objeto instancia que lo identifica entre todas las instancias de la misma clase.

2.2.4 Attribute ID: Identificador de Atributo. Es un valor entero de identificación asignado a una clase o atributo.

2.2.5 Service Code: Código de servicio. Es un valor entero de identificación que denota una solicitud de acción que puede estar dirigido a un objeto instancia particular u objeto clase.

2.2.6 Rango de direcciones:

Los siguientes términos se usan para definir rangos:

Abiertos: Es un rango de valores cuyos valores son definidos por la ODVA/CI y que son comunes por todos los participantes CIP.

Definición específica de fabricante (DEF): Es un rango de valores para un vendedor específico. Son usados por comercializadores para extender sus productos, más allá de las opciones que brindan las de direcciones abiertas.

Objeto instancia específico: Es un rango de valores cuyo significado es definido por un objeto instancia. Este rango aplica para códigos de servicio definidos.

Tabla 1: Rangos para Clase ID.

Rango	Definición
00-63 Hex	CIP común
64Hex-C7 Hex	DEF
C8 Hex-EF Hex	Reservado por la ODVA/CI para futuros usos
F0 Hex-2FF Hex	DEF
300 Hex-4FF Hex	Vendedor específico
500 Hex-FFFF Hex	Reservado por la ODVA/CI para futuros usos

Fuente: Traducido de[9]

Tabla 2: Rangos para códigos de servicio.

Rango	Definición
00-31Hex	CIP común. Para mensajes explícitos. Explicados en 2.3.2.3
32 Hex-4A Hex	DEF

4B Hex-63 Hex	Objeto instancia específico
64 Hex-7F Hex	Reservado por la ODVA/CI para futuros usos
80 Hex-FF Hex	Inválidos/No usados

Fuente: Traducido de[9]

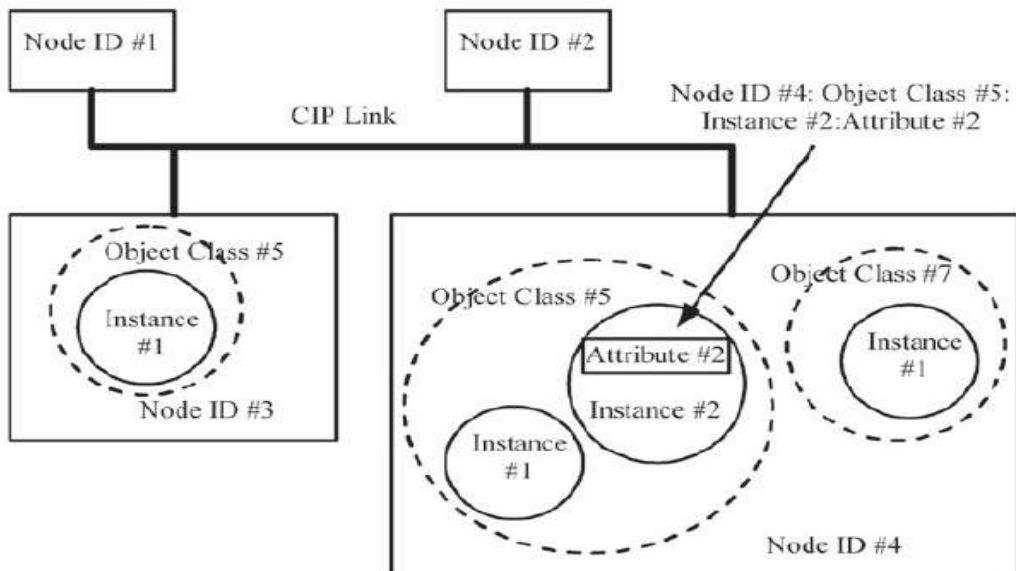
Tabla 3: Rangos para Atributos ID.

Rango	Definición
00-63Hex	CIP común.
64Hex-C7 Hex	DEF
C8 Hex-FF Hex	Reservado por la ODVA/CI para futuros usos
100 Hex-2FF Hex	CIP común.
300 Hex-4FF Hex	DEF
500 Hex-8 FF Hex	CIP común.
900 Hex-CFF Hex	DEF
D00 Hex-FFFF Hex	Reservado por la ODVA/CI para futuros usos

Fuente: Traducido de[9]

Tanto los objetos como los atributos específicos de fabricante deben seguir las pautas dadas por la especificación CIP.

Figura 3: Esquema de direccionamiento CIP.



Fuente: Tomado de [10]

Se puede observar en la figura 3 un ejemplo de una red CIP. La red posee cuatro nodos. Poniendo atención en el nodo #4, se puede observar que este posee 3 objetos instancia que se dividen en 2 clases de objetos. Siguiendo el esquema de direccionamiento definido por CIP, se puede vislumbrar la siguiente secuencia en el nodo #4:

Dirección de nodo #4; Clase Id #5: Identificador de Instancia #2: Identificador de atributo #2.

De esta forma, es inequívoca la aplicación requerida. Todos los nodos CIP deben ser direccionados siguiendo este esquema de direccionamiento.

2.3 MODELO DE OBJETOS CIP

Los objetos existentes en una aplicación CIP están definidos en los siguientes términos:

Tabla 4: Definiciones para un objeto.

Descripción	Una descripción que especifica la existencia de un objeto.
Código de clase	Un identificador hexadecimal asignado para cada objeto CIP.
Atributos	Los datos asociados con este objeto.
Servicios comunes	Una lista de servicios comunes definidos para este objeto.
Servicios para objetos de instancia específica	Las especificaciones completas de algún servicio único en este objeto.
Conexiones	Conexiones soportadas por este objeto
Comportamientos	La relación que hay entre valores de atributos y servicios.

Fuente: Traducido de[9]

2.3.1 Código de clase

Esta parte define un valor único hexadecimal para un objeto. Se usa el código de clase, para identificar la clase de objeto existente en un dispositivo. Las ODVA/CI son responsables para la designación y coordinación de los códigos de clase. Sin embargo, administrar los códigos de clase para DEF y garantizar los atributos que son únicos para estos; es responsabilidad de cada compañía asociada.

2.3.2 Atributos

Los atributos de un objeto, están divididos en 2 secciones:

-Atributos de clase.

-Atributos de instancia.

2.3.2.1 Atributos de clase:

Es un atributo que es mostrado por todos los objetos dentro de la misma clase. Los atributos de clase están definidos usando la siguiente terminología:

Tabla 5: Contenido atributos de clase

Identificador de atributo	Necesidad para implementarlo	Forma de acceso	NV	Nombre	Tipo de dato CIP	Descripción del atributo	Definición semántica del valor
1	2	3	4	5	6	7	8

Fuente: Traducido de[9]

1. Identificador de atributo: Es un valor de identificación de tipo entero asignado a un atributo. El identificador de atributo, identifica un atributo particular existente.

2. Necesidad para implementarlo: Especifica si el atributo es o no necesario en la implementación de la clase de objeto. Es decir un atributo puede ser *Requerido, Opcional o Condicional*.

Un atributo Condicional es solicitado si hay certeza de que el comportamiento del objeto y/o atributos, están implementados como definición de la clase o dentro del perfil de dispositivo.

Si un atributo de clase es Opcional, entonces se define un valor por defecto.

3. Forma de acceso: Especifica como una solicitud puede acceder a un atributo. Las definiciones para las formas de acceso son:

-Configurable (Set): El atributo puede ser accedidos por uno de los *set_atributes services*. Si el comportamiento del dispositivo no requiere uno de esos tipos de servicios, entonces no requiere ser implementado como configurable.

-Adquirible (Get): El atributo puede ser accedido por uno de los *get_atributes services*.

4. NV: Indica si el valor del atributo podrá ser mantenido a través de ciclos de encendido. Esta columna es usada en definiciones de objeto, en donde los valores de atributo son almacenados de forma no-volátil. Un valor “**NV**” indica que el valor puede ser recuperado. “**V**”significa que no puede ser recuperado.

5. Nombre: Referido al atributo.

6. Tipo de dato: Es usado en los *set_atributes* y *get_atributes* services. En ella se deberá mostrar el tipo de dato, para todos los participantes CIP usando la definición del atributo. Nombrados en 2.3.2.3.

7. Descripción del atributo: Provee información general acerca del atributo.

8. Definición semántica del valor: Especifica el significado del valor del atributo.

Importante: Hay 7 identificadores de atributos de clase, que son reservados para definiciones de clases de objetos. Estos son:

- Revisión.
- Máxima instancia.
- Número de instancias.
- Lista de atributos opcionales
- Lista de servicios opcionales.
- Máximo número de atributos de clase.
- Máximo número de atributos de instancia.

Como esos atributos son reservados, los números de identificadores de atributo del 1 al 7 siempre serán los mismos. Por lo tanto si se requiere adicionar un atributo de clase a una definición de objeto, se deberá comenzar con el identificador de clase #8. Los siete atributos de clase reservados, tienen las siguientes definiciones:

Tabla 6: Atributos de clase reservados para todas las definiciones de objetos de clase:

Identificador de atributo	Necesidad para implementar -lo	Forma de acceso	Nombre	Tipo de dato	Descripción del atributo	Definición semántica del valor
1	Condicional*	Get	Revisión.	UINT	Revisión de este objeto	El valor más popular asignado a este atributo es (01). Si se ha

						hecho una actualización a este valor entonces se incrementa en 1.
2	Opcional	Get	Máxima instancia.	UINT	Máximo número de instancias de objeto creadas recientemente	El máximo número de instancias de objeto creadas.
3	Opcional	Get	Numero de instancias	UINT	Numero de instancias de objeto creadas recientemente	El número de instancias de objeto.
4	Opcional	Get	Lista de atributos opcionales	Estructura de	Lista de atributos de instancia utilizadas en la implementación de una clase de objeto	Una lista de números específicos de atributo, de los atributos opcionales implementados en un dispositivo para esta clase.
				UINT	Numero de atributos en la lista de atributos opcionales	El número de la cantidad de atributos en la lista
				Array de UINT	Lista de numero de atributos opcionales	El número de atributos opcionales.
5	Opcional	Get	Lista de servicios opcionales	Estructura de	Lista de servicios opcionales utilizados en la implementación de una clase de objeto	Una lista de códigos de servicio de los servicios opcionales implementados en un dispositivo para esta clase.
				UINT	Numero de servicios en la lista de servicios opcionales	El número de código de servicios en la lista.
				Array de UINT	Lista de códigos de servicio opcionales	Los códigos de servicios opcionales
6	Opcional	Get	Máximo número de atributos de clase.	UINT	El número de identificadores de atributo de clase implementadas en un dispositivo para esta clase.	
7	Opcional	Get	Máximo número de atributos de instancia.	UINT	El número de identificadores de atributo de instancia implementada en un dispositivo para esta clase.	

Fuente: Traducido de[9]

* Si el valor es 01, entonces este atributo es OPCIONAL en implementación. Si el valor es más grande que 1, entonces este atributo es requerido.

Nota: La definición de los tipos de datos, se encuentra en la tabla 50.

2.3.2.2 Atributos de instancia:

Un atributo de instancia es un atributo que es único en una instancia de objeto y no hace parte de una clase objeto.

Estos atributos están definidos en los mismos términos que los atributos de objeto.
No hay atributos de instancia reservados.

Tabla 7: Contenido atributos de instancia

Identificador de atributo	Necesidad para implementarlo	Forma de acceso	NV	Nombre	Tipo de dato CIP	Descripción del atributo	Definición semántica del valor
1	2	3	4	5	6	7	8

Fuente: Traducido de[9]

2.3.2.3 Servicios:

Los códigos de servicio son utilizados para definir la acción solicitada cuando un objeto es direccionado a través de mensajería explícita.

Aparte de las funciones de lectura y escritura, se ha definido un conjunto de servicios CIP.

Estos servicios CIP son de naturaleza común, lo que significa que se pueden utilizar en todas las redes CIP y son útiles para una variedad de objetos.

También, existen códigos de servicio específicos de objetos que pueden tener diferentes significados para el mismo código, dependiendo de la clase a la cual pertenece el objeto.

Además de los códigos de servicio generales y de los códigos específicos de objetos definidos por CIP, los desarrolladores de dispositivos pueden definir otros servicios específicos para sus productos. Aunque esta característica otorga mucha flexibilidad para los fabricantes, tiene como desventaja que los servicios específicos de fabricante no pueden ser entendidos universalmente.

Los fabricantes deben entregar, al menos, una descripción pública de estos servicios particulares a sus clientes que necesiten acceder a esta información.

Los componentes de un servicio común son los siguientes:

Tabla 8: Contenido de un servicio común

Servicio	Necesidad para implementarlo		Servicio	Descripción del
Código	Clase	Instancia	Nombre	Servicio
1	2	3	4	5

Fuente: Traducido de[9]

1. Código de servicio: Es un valor hexadecimal asignado para servicio CIP.

2-3 Necesidad para implementarlo: Especifica si un servicio tiene la necesidad o no para implementarlo para un nivel de **Clase** o de **Instancia**. En esta columna puede aparecer una de estas 4 especificaciones:

- Opcional.
- Requerido.
- Condicional
- No aplicable (“n/a”).

Un servicio Opcional se necesita si hay certeza de que los comportamientos de un objeto, atributos y/o los servicios están implementados como una definición para un objeto. Todos los servicios condicionales especifican las condiciones necesarias para que el servicio sea pretendido. Si es No aplicable, no será soportado para un nivel de **Clase** o de **Instancia**.

Importante: Si los servicios opcionales implementados en una clase, y la lista de servicios de atributos de clase opcionales esta también implementado en la clase, el servicio será incluido en la lista de servicios.

Los servicios desencadenan el comportamiento de un objeto basado en los valores de atributo por el servicio. Los servicios pueden ser directamente accedidos por un nivel de clase o un nivel de objeto de instancia, el cual puede producir diferentes comportamiento para cada nivel.

- **Nivel de clase:** Comportamiento desencadenado por los servicios enviados a una clase de objetos.
- **Nivel de instancia:** Comportamiento desencadenado por los servicios enviados a una instancia de un objeto.

4. Nombre de servicio: Los códigos y nombres de los servicios CIP son mostrados a continuación:

Tabla 9: Nombres y códigos de servicios

Código de servicio (en hexadecimal)	Nombre de servicio
00	Reservado para usos futuros.
01	Get_Attributes_All
02	Set_Attributes_All Request
03	Get_Attributes_List

04	Set_Attributes_List
05	Reset
06	Start
07	Stop
08	Create
09	Delete
0A	Multiple_Service_Packet
0B-0C	Reservados para futuros usos
0D	Apply_Attributes
0E	Get_Attribute_Single
0F	Reservado para futuros usos
10	Set_Attribute_Single
11	Find_Next_Object_Instance
12-13	Reservados para futuros usos
14	Error Response (usado solamente para DeviceNet)
15	Restore
16	Save
17	No Operation (NOP)
18	Get_Member
19	Set_Member
1 ^a	Insert_Member
1B	Remove_Member
1C	GroupSync
1D-31	Reservado para adicionar servicios.

Fuente: Traducido de[9]

5. Descripción del servicio: Provee una definición breve del servicio.

2.4 LIBRERÍA DE OBJETOS DE LAS REDES CIP

El conjunto de redes que implementan el protocolo CIP, posee una amplia colección de objetos definidos comunes. El conjunto total de clases de objeto se puede subdividir en tres tipos:

General-use (o de uso general).

Application-specific (o específico de aplicación).

Network-specific (o específico de red).

Los objetos definidos en el volumen 1 de la Librería de Redes CIP están disponibles para utilizarse en todas las adaptaciones de la red CIP. Algunos de estos objetos pueden requerir cambios o limitaciones específicas cuando están implementadas en algunas de las adaptaciones de red. Estas excepciones se observan en el volumen de especificaciones correspondiente a la red.

Por lo tanto, para ver el cuadro completo de implementación de una red en particular, se debe considerar el volumen 1 y el volumen particular de la red (ya habíamos mencionado que, Ethernet/IP se compone del Volumen 1 y del Volumen 2 de CIP).

TABLA 10 Objetos definidos por CIP para uso general

OBJETOS DE USO GENERAL CIP	
Identity	Message Router
Connection Manager	Parameter
Register	Assembly
Acknowledge Handler	Connection
Connection Configuration	Port
File Selection	Connection Configuration
Parameter Group	

Fuente: Traducido de[9]

TABLA 11 Objetos específicos de aplicación

Objetos específicos de aplicación	
AC/DC Drive	Overload
Analog Group	Position Controller
Analog Input Group	Position Controller
Supervisor	Analog Output Group
Analog Input Point	Position Sensor
Presence Sensing	S-Analog Actor
Analog Output Point	S-Analog Sensor
Block Sequencer	S-Device Supervisor
Command Block	S-Gas Calibration
Control Supervisor	S-Partial Pressure
Discrete Group	Discrete Input Group
Discrete Output Group	S-Single Stage Controller
Softstart Starter	Discrete Input Point
Discrete Output Point	Group Trip Point
Motor Data	Safety Supervisor
Safety Validator	Softstart Starter
Trip Point	

Fuente: Traducido de[9]

Tabla 12 Objetos específicos de red CIP

Objetos específicos de red
ControlNet
ControlNet Keeper
ControlNet Scheduling
DeviceNet
Ethernet Link
TCP/IP Interface

Fuente: Traducido de[9]

Los objetos de uso general pueden ser encontrados en muchos dispositivos diferentes, mientras que los objetos específicos de aplicación típicamente se encuentran sólo en los dispositivos diseñados para tal aplicación.

2.4.1 Conjunto mínimo de objetos requeridos en un dispositivo:

Aunque son varios los tipos de objetos definidos para las redes CIP, un dispositivo típico sólo implementa un subconjunto de todos ellos. De modo que para que un dispositivo puede operar en una red CIP debe implementar como mínimo determinados objetos.

Los objetos requeridos para un típico dispositivo CIP (ya sea DeviceNet, ControlNet o EtherNet/IP) son:

Un **Connection-Objetc** (u objeto de conexión), o bien un **Connection Manager Object** (u objeto gestor de conexión). Estos objetos pertenecen al grupo de objetos de usogeneral y cumplen las funciones de objetos de comunicación.

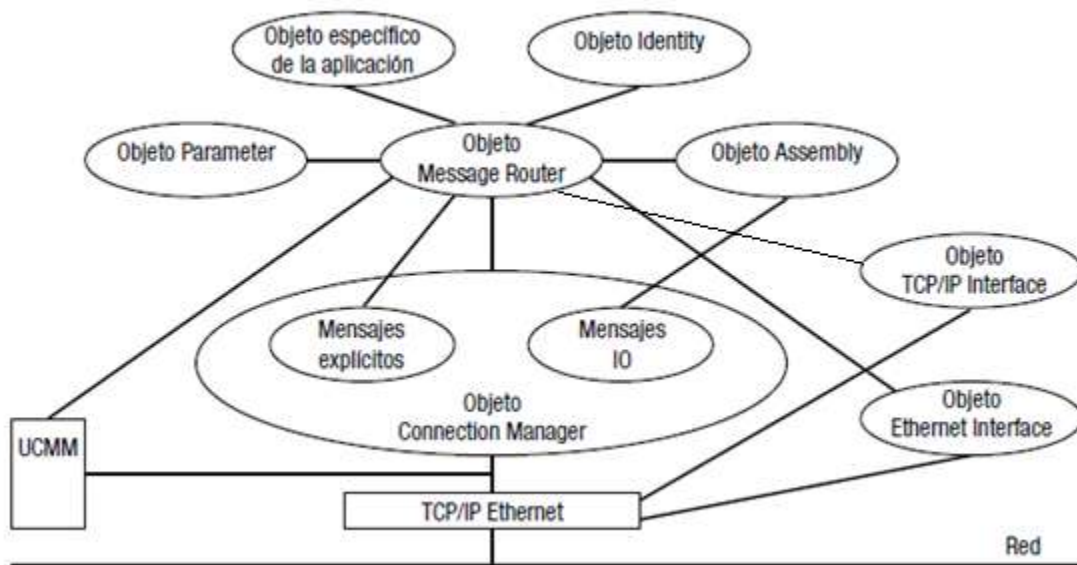
Un **Identity-Object**, u objeto identidad. Éste objeto pertenece al grupo de objetos de uso general.

Un **Message Router Object**, u objeto ruteador de mensaje (o al menos implementar esta función). Este objeto pertenece al grupo de objetos de uso general.

Uno o más **objetos específicos de red**. Estos objetos no son del grupo de uso general, sino que dependen de la red.

A continuación se representa un dispositivo típico de acuerdo con el modelo abstracto de objeto.

FIGURA 4: Dispositivo típico representado según el modelo de objetos CIP.



Fuente: Tomada de [11], 28 de Septiembre de 2012.

Este subconjunto de objetos es requerido en todo dispositivo para que éste sea funcional.

Es claro que el dispositivo debe incorporar, además, el objeto de aplicación específica según el tipo de dispositivo que sea. Estos objetos requeridos en todo dispositivo típico aparecen sombreados en la representación de un dispositivo con el modelo de objetos que se entrega en la figura 3.

2.5 PERFILES DE DISPOSITIVO

Es posible diseñar productos para las redes CIP siguiendo solamente las especificaciones de comunicación de la red y el modelo de objetos. Sin embargo esto resultaría en que productos similares finalmente tendrían estructuras y comportamientos muy diferentes unos de otros. Con el objetivo de superar esta situación y permitir una aplicación de dispositivos CIP más simple, se han agrupado los dispositivos de funcionalidad similar y se les ha asociado un determinado perfil.

Este *perfil de dispositivo* CIP contiene la descripción completa de la estructura y el comportamiento del objeto.

En la siguiente tabla se muestran los perfiles de dispositivos, junto a su número identificativo, definidos en el Volumen 1 de CIP.

Tabla 13 Perfiles de dispositivos CIP

PERFILES DE DISPOSITIVO CIP (código hexadecimal)	
AC Drives Device (02)	Mass Flow Controller (1A)
Communications Adapter (0C)	Motor Overload Device (03)
Contactora (15)	Motor Starter (16)
ControlNet Phy. Layer Component (32)	Photoelectric Sensor (06)
ControlNet PLC (0E)	Pneumatic Valve (1B)
DC Drives (13)	Position Controller (10)
DC Power Generator (1F)	Process Control Valve (1D)
Encoder (22)	Residual Gas Analyzer (1E)
Fluid Flow Controller (24)	Resolver (09)
General Purpose Discrete I/O (07)	RF Power Generator (20)
Generic Device (00)	Safety Discrete I/O (23)
Human Machine Interface (18)	Softstart Starter (17)
Inductive Proximity Switch (05)	Turbomolecular Vacuum Pump (21)
Limit Switch (04)	Vacuum/Pressure Gauge (1C)

Fuente: Traducida de [9]

Los desarrolladores de dispositivos deben usar un perfil. Cualquier dispositivo que no cae dentro de uno de los perfiles especializados enlistados en la tabla 13 debe usar un perfil de dispositivo genérico o un perfil específico de fabricante. En la documentación provista por el fabricante al usuario, deben ser descritos el perfil que utiliza y qué partes son implementadas.

Todo perfil de dispositivo se compone un conjunto de objetos, tanto requeridos como opcionales, y un comportamiento asociado con el tipo de dispositivo particular.

Además de los objetos de definición pública, los vendedores pueden agregar sus propios objetos si sus dispositivos entregan funcionalidades adicionales.

Asimismo, los fabricantes pueden desarrollar sus propios perfiles de dispositivos en el contexto de perfil específico de vendedor.

En la tabla 14 se enlistan los rangos asignados de los números que identifican los perfiles públicos (es decir, los definidos por ODVA y ControlNet Internacional) y los específicos de vendedor.

Tabla 14 Direcciones para perfiles de dispositivo

Tipo	Rango	Cantidad
Perfiles de dispositivos públicos	00 Hex-63 Hex	100
Perfiles específicos de fabricante	64 Hex - C7 Hex	100
Reservado por CIP.	C8 Hex – FF Hex	56
Perfiles específicos de fabricante	100 Hex - 2FF Hex	512
Vendedor específico	300 Hex - 4FF Hex	512
Reservado por CIP.	500 Hex – FFFFhex	64256

Fuente: Traducida de [9]

2.5.1 Perfil de dispositivo para Ethernet/IP:

Generalmente en la industria, todo dispositivo Ethernet/IP implementa alguna instancia de los siguientes objetos:

- Objeto Identity (código de clase=01 Hex).
- Objeto TCP/IP Interface (código de clase=F5 Hex).
- Objeto Ethernet Link (código de clase=F6 Hex).
- Objeto Message Router (código de clase=02 Hex).
- Objeto Connection Manager (código de clase=06 Hex)

2.5.1.1 Descripción de los atributos del objeto identity:

Identificador de clase: 01 Hex

Este objeto provee información detallada acerca del dispositivo. El objeto identity está presente en todos los productos CIP.

Este y todos los objetos CIP, pueden implementan atributos de clase, los cuales fueron explicados en el ítem 2.3.2.1.

Nota: De aquí en adelante describiremos solamente atributos de instancia, que son requeridos por un objeto y serán implementados en este proyecto.

Tabla 15 Atributos de instancia del objeto identity.

Identificador de atributo	Necesidad para implementarlo	Forma de acceso	Nombre	Tipo de dato CIP	Descripción del atributo	Definición semántica del valor
1	Requerido	Get	Vendor	UINT	Identificación de cada	Ver sección

			ID		fabricante por numero	"semántica"
2	Requerido	Get	Device type	UINT	Identificación de un tipo general de producto	Ver sección "semántica"
3	Requerido	Get	Product code	UINT	Identificación de un producto particular de un fabricante específico	Ver sección "semántica"
4	Requerido	Get	Revision	Estructura de:	Revisión de los ítems del objeto identity	
			Major revision	USINT		Ver sección "semántica"
			Minor revision	USINT		Ver sección "semántica"
5	Requerido	Get	Status	WORD	Resumen del status del dispositivo	Ver sección "semántica"
6	Requerido	Get	Serial Number	UDINT	Número de serie del dispositivo	Ver sección "semántica"
7	Requerido	Get	Product Name	SHORT_STRING	Identificación en ASCII del producto	Ver sección "semántica"

Fuente: Traducida de [9]

Definición semántica de los valores de atributo del objeto identity:

2.5.1.1.1 Vendor ID: El atributo Vendor ID identifica al fabricante del dispositivo. Corresponde a un valor entero asignado por ODVA o ControlNet Internacional al fabricante que lo solicite. Este valor será el mismo en cada red.

2.5.1.1.2 Device Type: El atributo Device Type, especifica qué perfil se ha utilizado para este dispositivo. Debe ser uno de los Device Type descritos por CIP o por un tipo específico de fabricante.

2.5.1.1.3 Product Code: El atributo Product Code, es un número entero definido por el fabricante del dispositivo. Este código se utiliza para distinguir múltiples productos del mismo Device Type y del mismo fabricante.

2.5.1.1.4 Revision: El atributo Revision, está dividida en dos valores enteros que especifican una Major Revision, o Revisión Principal, y una Minor Revision, o Revisión Menor. Cualquier cambio (o cambios) del dispositivo que dé lugar a modificar su comportamiento en la red debe ser reflejado en un cambio (o cambios) en la parte Minor del atributo Revisión. Cualquier cambio (o cambios) en el dispositivo que requiere una revisión de su Electronic Data Sheet, EDS (Hoja de Datos Electrónica en español) debe ser reflejado en un cambio en la parte Major del atributo Revisión. Los atributos Vendor ID, Device Type, Product Code y Major Revision proporcionan una identificación inequívoca de un EDS para este dispositivo.

2.5.1.1.5 Status: El atributo Status proporciona la información sobre el estado del dispositivo, por ejemplo si está siendo controlado por otro dispositivo o configurado, y si ha ocurrido alguna falla menor o importante.

El atributo status es una palabra, con las siguientes definiciones para bit.

Tabla 16: Definición de bits para el atributo de instancia Status, del objeto identity

Bit(s)	Denominado	Definición
0	Propietario	VERDADERO indica que el dispositivo (o un objeto dentro del dispositivo) tuvo un propietario. Dentro del paradigma Maestro/esclavo la configuración del significado de los bits que predefine la conexión de la configuración ha sido asignada por un maestro. Fuera del paradigma Maestro/esclavo el significado del bit es un valor no asignado por la IEEE.
1		Reservado, será 0
2	Configurado	VERDADERO indica que la aplicación del dispositivo ha sido configurada para algo diferente fuera de los límites tradicionales. Este no incluirá configuración en las comunicaciones.
3		
4-7	Status extendido de dispositivo	La EDS se indicará si el dispositivo sigue la definición pública de estos bits utilizando la palabra clave DeviceStatusAssembly en la sección [Device] de la EDS. Si estos bits son específicos del fabricante, entonces, se enumeran en la EDS utilizando parámetros Assembly yParameter.
8	Falla menor recuperable	VERDADERO indica que el dispositivo a detectado un problema el cual por si solo puede recuperarse. El problema no causa que el dispositivo caiga dentro de uno de los estados de falla. Ver sección 5-2.6 Volumen 1.
9	Falla menor irrecuperable	VERDADERO indica que el dispositivo a detectado un problema el cual no podrá recuperarse. El problema no causa que el dispositivo caiga dentro de uno de los estados de falla. Ver sección 5-2.6 Volumen 1.
10	Falla mayor recuperable	VERDADERO indica que el dispositivo el cual lo va a llevar al estado de "Falla mayor recuperable". Ver sección 5-2.6 Volumen 1.
11	Falla mayor irrecuperable	VERDADERO indica que el dispositivo el cual lo va a llevar al estado de "Falla mayor irrecuperable". Ver sección 5-2.6 Volumen 1.
12-15		Reservado, será 0

Fuente: Traducida de [9]

2.5.1.1.6 Serial Number: El atributo serial number, o número de serie, se utiliza para identificar únicamente los dispositivos individuales en conjunto con el Vendor ID, es decir, ninguno de dos dispositivos CIP de un determinado fabricante puede tener el mismo número de serie.

2.5.1.1.7 Product Name: El atributo Product Name, o Nombre de Producto, permite al fabricante darle un nombre significativo en código ASCII, de hasta 32 caracteres, al dispositivo.

Tabla 17 Servicios soportados por el objeto identity

Código de servicio	Necesidad para implementar		Nombre del servicio	Descripción del servicio
	Clase	Instancia		
0E Hex	Opcional	Requerido	Get_Attribute_Single	Este servicio devuelve el valor del atributo de la instancia especificado.
01 Hex	Opcional	Condicional	Get_Attributes_All	Este servicio devuelve el valor de todos los atributos de la instancia.
10 Hex	n/a	Condicional	Set_Attribute_Single	Modifica un atributo

Fuente: Traducida de [9]

2.5.1.2 Descripción de los atributos del objeto TCP/IP interface:

Identificador de clase: F5 Hex

El objeto de interfaz TCP/IP provee el mecanismo para configurar la dirección IP del dispositivo.

TABLA 18: Atributos de instancia del objeto TCP/IP Interface

Identificador de atributo	Necesidad para implementarlo	Forma de acceso	Nombre	Tipo de dato CIP	Descripción del atributo	Definición semántica del valor
1	Requerido	Get	Status	Dword	Estado de la configuración	Ver sección "semántica"
2	Requerido	Get	Configuration Capability	Dword	Marcas para la capacidad de la configuración	Ver sección "semántica"
3	Requerido	Get	Configuration Control	Dword	Marcas para el control de la configuración	Ver sección "semántica"

4	Requerido	Get	Physical Link Object	Epath	Ruta electrónica al objeto de la conexión física	Ver sección "semántica"
5	Requerido	Get	Interface Configuration	Estructura de:	Interfaz de configuración de la red TCP/IP	Ver sección "semántica"
			Ip address	Udint	La dirección IP del dispositivo	Ver sección "semántica"
			Network Mask	Udint	La máscara de subred del dispositivo	Ver sección "semántica"
			Gateway Address	Udint	La dirección de pasarela por defecto	Ver sección "semántica"
			Name server	Udint	Nombre del servidor primario	Ver sección "semántica"
			Name server 2	Udint	Nombre del servidor secundario	Ver sección "semántica"
			Domain name	String	Nombre del dominio	Ver sección "semántica"
6	Requerido	Get	Host name	String	Nombre del host	Ver sección "semántica"

Fuente: Traducida de [12]

Definición semántica de los valores de atributo del objeto TCP/IP interface:

2.5.1.2.1 Status: El atributo status, es un mapa de bits que indica el estado de la red TCP/IP.

Tabla 19: Atributo Status

Bit(s)	Descripción
0-3	<p>0: El atributo interfaz de configuración no ha sido configurado.</p> <p>1: La interfaz de configuración se ha configurado satisfactoriamente a través de DHCP, BOOTP.</p> <p>2: La interfaz de configuración ha sido configurado, por la programación de un hardware</p> <p>3-15 : Reservado para futuros usos.</p>

Fuente: Traducida de [12]

2.5.1.2.2 Configuration capability: El atributo Configuration capability es un mapa de bits, que determina si el dispositivo puede obtener una configuración inicial por una red opcional.

Tabla 20: Atributo Configuration capability

Bit(s)	Descripción
0-4	<p>0: Indica que el dispositivo es capaz de obtener una configuración vía BOOTP.</p> <p>1: Indica que el dispositivo es capaz de resolver los nombres de host por medio del servidor DNS.</p> <p>2: Indica que el dispositivo es capaz de obtener una configuración vía DHCP.</p> <p>3:(DHCP-DNS) Indica que el dispositivo está en capacidad de enviar este Host name en la solicitud DHCP como documento en internet draft <draft-ietf-dhc-dhcp-dns-12.txt>.</p> <p>4: Indica que la interfaz de configuración es manipulable. Algunos dispositivos como PC o una estación de trabajo pueden dejar su configuración por medio del objeto TCP/IP Interface</p>

Fuente: Traducida de [12]

2.5.1.2.3 Configuration Control: Este atributo, es un mapa de bits que indica el método implementado para obtener su configuración de la dirección IP.

Tabla 21: Atributo Configuration control

Bit(s)	Descripción
0-3	0: El dispositivo usa el valor la interfaz de configuración previamente almacenada (por ejemplo en memorias no volátiles o vía Switches de hardware, etc)
4	
5-31	<p>1: Indica que el dispositivo obtuvo su configuración vía BOOTP.</p> <p>2: Indica que el dispositivo obtuvo su</p>

	<p>configuración vía DHCP</p> <p>3-15: Reservado para futuros usos.</p> <p>4: Indica que el dispositivo resuelto los nombres de host por medio del servidor DNS.</p> <p>5-31: Reservado para futuros usos.</p>
--	--

Fuente: Traducida de [12]

2.5.1.2.4 Physical Link Object: Este atributo devuelve la ruta electrónica al objeto de la conexión física, el cual es la clase conexión Ethernet. Hay dos componentes del atributo: un Path Size (en UINTS) y un Path. El Path contiene un Logical Segment, tipo clase, y un Logical Segment, tipo instancia que identifica la ruta física del objeto. El tamaño máximo del Path Size es 6 (asumiendo un Logical Segment de 32 bits por cada clase e instancia).

Si el puerto CIP asociado con el objeto TCP/IP interface tiene una conexión Ethernet, este punto será asociado con un atributo del objeto Ethernet Link (código de clase F6 hex).

Por ejemplo la ruta que sigue especifica la instancia 1 del objeto de la Conexión Ethernet (0x20 0xF6 0x24 0x01). En la sección 2.7 explicaremos como funciona los DATA MANAGEMENT.

2.5.1.2.5 Interface Configuration: Este atributo contiene todos los parámetros requeridos para operar en un nodo TCP/IP.

IP address: Contiene la dirección IP del dispositivo.

Network Mask: Contiene la máscara de subred.

Gateway address: Contiene la dirección de la pasarela.

Name server: Contiene la dirección IP del nombre del servidor. El nombre del servidor es usado para resolver Host Names.

Name server 2: Contiene la dirección IP del segundo nombre del servidor. Es usado cuando el nombre del servidor no está avalado, o es incapaz de resolver un Host Name.

2.5.1.2.6 Host Name: Contiene el nombre de anfitrión que aparece en la red, cuando se utiliza DHCP.

Por ejemplo, si por defecto el host name es “Odva.org”, y el dispositivo necesita resolver un host name como “PLC”, entonces el dispositivo intenta resolver el host name como “plc.odva.org”.

Tabla 22: Servicios soportados por el objeto TCP/IP interface

Código de servicio	Necesidad para implementar		Nombre del servicio	Descripción del servicio
	Clase	Instancia		
0E Hex	Opcional	Requerido	Get_Attribute_Single	Este servicio devuelve el valor del atributo de la instancia especificado.
01 Hex	Opcional	Condicional	Get_Attributes_All	Este servicio devuelve el valor de todos los atributos de la instancia.

Fuente: Traducida de [12]

2.5.1.3 Descripción de los atributos del objeto Ethernet Link:
Identificador de clase: F6 Hex

El objeto de conexión Ethernet sigue la información de configuración y de diagnóstico del puerto Ethernet.

Tabla 23: Atributos de instancia del objeto Ethernet Link

Identificador de atributo	Necesidad para implementarlo	Forma de acceso	Nombre	Tipo de dato CIP	Descripción del atributo	Definición semántica del valor
1	Requerido	Get	Interface Speed	Udint	Esta instancia devuelve la velocidad de la interfaz. Depende la velocidad de la Ethernet.	Velocidad en Mbps(0,10, 100, etc)
2	Requerido	Get	Interface Flags	Dword	Indicador para recoger	Ver sección “semántica”

					información de la interfaz:	
3	Requerido	Get	Physical Address	Array de 6 Usints	Este atributo devuelve la dirección MAC de 6 bytes.	La dirección Ethernet será asignada por el fabricante y será única para los requerimientos de la IEEE 802.3

Fuente: Traducida de [12]

Definición semántica de los valores de atributo del objeto Ethernet Link:

2.5.1.3.1 Interface Flags: Estos indicadores recogen información de la interfaz:

-*Bit 0 (estado de la conexión):* El estado de la conexión indica cuándo la interfaz de comunicaciones Ethernet 802.3 está conectada a una red activa. 0 indica una conexión inactiva. 1 indica una conexión activa.

-*Bit 1 (half/full duplex):* 0 indica que la interfaz se está ejecutando en half dúplex. 1 indica que la interfaz se está ejecutando en full dúplex.

-*Bits 2...31 (reservado):* Estos bits se establecen en 0.

Tabla 24 Servicios soportados por el objeto Ethernet Link

Código de servicio	Necesidad para implementar		Nombre del servicio	Descripción del servicio
	Clase	Instancia		
0E Hex	Opcional	Requerido	Get_Attribute_Single	Este servicio devuelve el valor del atributo de la instancia especificado.
01 Hex	Opcional	Condicional	Get_Attributes_All	Este servicio devuelve el valor de todos los atributos de la instancia.

Fuente: Traducida de [12]

2.5.1.4 Descripción del objeto Message Router:

Identificador de clase: 02 Hex

El objeto Message Router dirige mensajes explícitos entre objetos de un dispositivo.

Para esto, CIP define un formato estándar para la entrega de datos a través del objeto Message Router.

Nota: Este objeto maneja atributos de instancia opcionales.

Tabla 25: Formato de solicitud del Message Router

Nombre del parámetro	Tipo de dato	Descripción
Servicio	Usint	Código del servicio de la solicitud.
Tamaño del Path* de la solicitud	Usint	El número de palabras de 16 bits del path de la solicitud (siguiente elemento)
Path de la solicitud	Packet EPATH	Este es un array el cual contiene el path de la solicitud (Clase ID, Instancia ID, etc) para esta transacción.
Datos de la solicitud	Array de octeto	Datos específicos del servicio entregados en la solicitud del mensaje explícito. Si no hay datos adicionales en la solicitud del mensaje explícito. Entonces este array estará vacío.

Fuente: Traducida de [9]

* Ver sección 2.9 Data Management

Tabla 26: Formato de respuesta del Message Router

Nombre del parámetro	Tipo de dato	Descripción
Réplica del servicio	Usint	Código de la réplica del servicio.
Estado general	Usint	Uno de los códigos de estado general CIP. Ver sección 2.9.

Tamaño del estado adicional	Usint	El número de palabras de 16 bits en el array de estado adicional.
Estado adicional	Array de palabras	Estado adicional
Datos de respuesta	Array de octeto	Datos de respuesta de la solicitud o datos de error adicional si el estado general indica un error.

Fuente: Traducida de [9]

El objeto Message Router solo tiene atributos opcionales, pero es de vital importancia en todo dispositivo CIP ya que:

- Interpreta la instancia de clase especificada en un mensaje.
- Rutea un servicio al objeto específico.
- Rutea una respuesta correcta a la fuente de origen.

2.5.1.4.1 Solicitud de servicio:

-La interpretación de la instancia de clase se realiza en cada uno de los servicios recibidos por el mensaje

-Cualquier instancia de la clase que no puede ser interpretada en la implementación de un dispositivo del Message Router, reportara un error de objeto no encontrado.

- El servicio está dirigido a un objeto de destino.

2.5.1.4.2 Solicitud de respuesta:

Todos los servicios de respuesta se dirige a la conexión de mensaje explícita a través del cual la petición de servicio fue recibida

2.5.1.5 Descripción del objeto Connection Manager:

Identificador de clase: 02 Hex

Este Objeto tiene como objetivo asignar y gestionar los recursos internos relacionados con I/O y las conexiones de mensajería explícita de tipo (UCMM).

Nota: Este objeto maneja atributos de instancia opcionales.

La importancia de este objeto radica en sus servicios específicos de objeto.

Tabla 27: Servicios específicos de objeto del Connection Manager

Código del servicio (en hex)	Necesidad para implementarlo	Nombre del servicio	Descripción del servicio
4E	Condicional	Forward_Close	Cierra una conexión
52E	Condicional	Unconnected_send	Solamente dispositivos <i>originator</i> y dispositivos que ruten entre enlaces podrán implementarlo
54E	Condicional	Forward_Open	Abre una conexión, de un tamaño máximo de 511 bytes.
56E	Opcional	Get_Connection_Data	Para diagnósticos de una conexión.
57E	Opcional	Search_Connection_Data	Para diagnósticos de una conexión.
59E	Obsoleto		
5A	Condicional	Get_Connection_owner	Determinar el propietario de una conexión redundante
5B	Opcional	Large_Forward_Open	Abre una conexión, de un tamaño máximo de 65535 bytes

Fuente: Traducida de [9]

2.6 Adaptación de la red Ethernet/IP a CIP

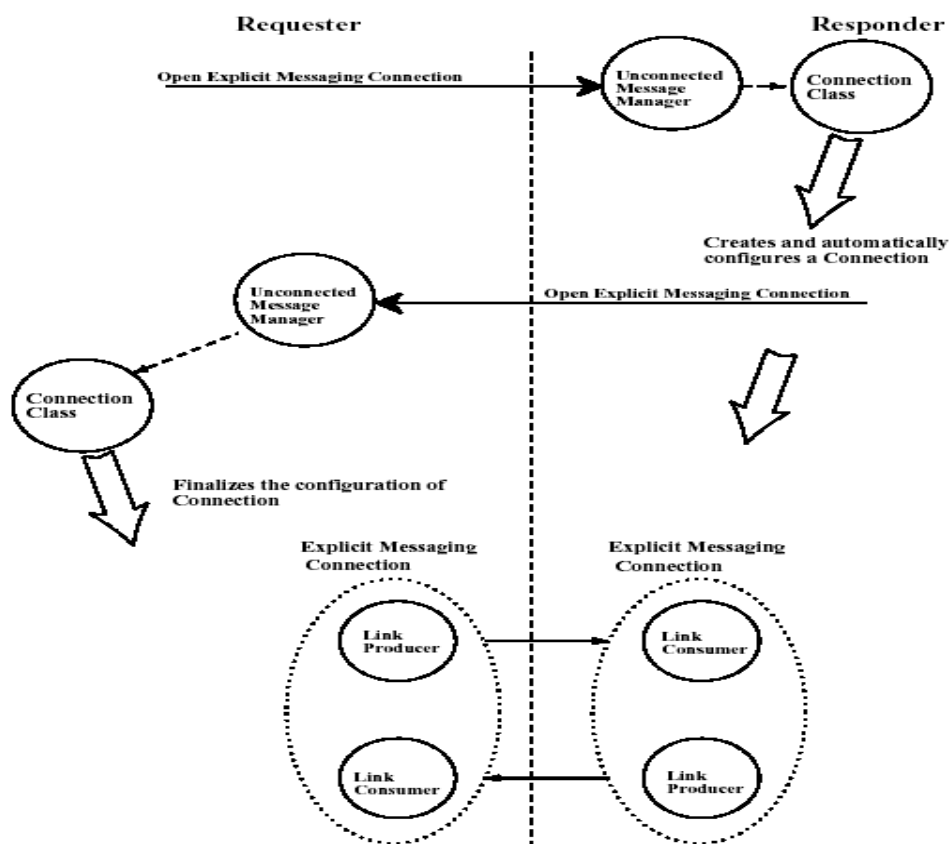
Puesto que el intercambio de información en una Red CIP se hace a través de conexiones, se ha definido un proceso para establecerlas cuando los dispositivos aún están desconectados.

Esto se hace a través de la función *Unconnected Message Manager (UCMM)*, que puede traducirse al español como el Gestor de Mensajería No Conectada,

que es responsable del establecimiento tanto de mensajería explícita sin conexión como de conexiones I/O.

Una solicitud UCMM contiene toda la información requerida para crear una conexión entre el dispositivo originador (**originator**) y el dispositivo objetivo (**target**) y, si es requerido, una segunda conexión entre el dispositivo objetivo y el dispositivo originador.

FIGURA 5: Pasos para el establecimiento de una conexión explícita (sin conexión).



Fuente: Tomada de [9]

La conexión de mensajería explícita es incondicionalmente punto a punto. La conexión punto a punto existe en dos dispositivos solamente. El dispositivo que solicita la conexión (el cliente) es un terminal de la conexión y el módulo que recibe y contesta la solicitud (el servidor) es el otro terminal.

EtherNet/IP está basado enteramente en las tecnologías TCP/IP y UDP/IP y usa esos principios sin ninguna modificación. TCP/IP es usado básicamente para el

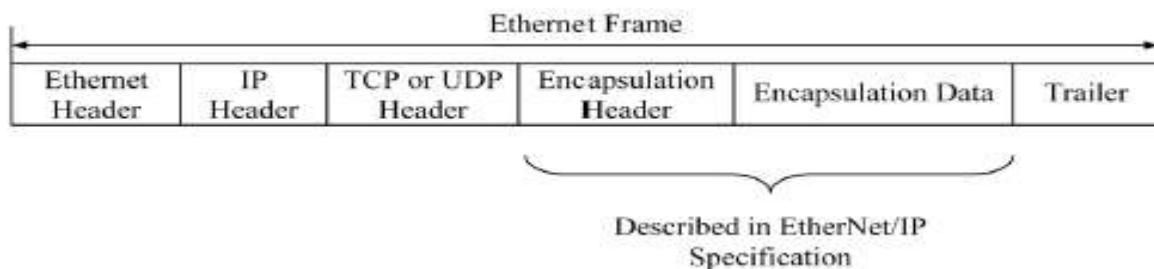
uso de transmisión de mensajería explícita, mientras que UDP/IP es usada para conexiones I/O.

La encapsulación del protocolo define dos números de puerto reservados TCP. Todos los dispositivos Ethernet/IP aceptan al menos dos conexiones sobre el puerto TCP AF12 hex. Este puerto es usado para todos los TCP basado en mensajes explícitos, ya sea conectado o no conectado. Una vez la conexión TCP por el puerto AF12 es establecida, todos los datos enviados a través de la trama TCP será de la forma de la sección 2.6.2

2.6.1 Uso general de la trama Ethernet:

Todo mensaje encapsulado CIP sobre una red EtherNet/IP está basado en una trama Ethernet con su respectiva cabecera IP.

FIGURA 6: Relación entre CIP y la trama Ethernet



Fuente: Traducida de [12]

Las cabeceras Ethernet, IP y TCP o UDP están descritas a través de estándares internacionales, por lo tanto los usos de esas cabeceras serán nombrados en la especificación Ethernet/IP para entender cómo serán usados.

2.6.2 Encapsulamiento de mensajes

2.6.2.1 Estructura de encapsulamiento:

Todos los mensajes encapsulados, vía TCP o UDP serán por el puerto AF12. Estará compuesto por un tamaño de longitud fijo de 24 bytes seguido por una porción de datos opcionales. El tamaño de longitud del mensaje (incluido la cabecera) estará limitado a 65535 bytes. Esta estructura es como lo muestra la siguiente figura.

Tabla 28: Paquete de encapsulación.

Estructura	Nombre del campo	Tipo de dato	Significado del campo
Cabecera de encapsulación	Command	Uint	Comando de encapsulación
	Lenght	Uint	Longitud, en bytes, de la porción de datos del mensaje
	Session Handle	Udint	Sesion de identificación (aplicación dependiente)
	Status	Udint	Código de estado
	Sender Context	Array de octetos	Información pertinente solamente al envío de un comando de encapsulación
	Options	Udint	Distintivos de opciones
Comando de datos específicos	Encapsulated data	Array de 0 a 65611 octetos.	La porción de datos de encapsulación del mensaje es requerido solamente para ciertos comandos

Fuente: Traducida de [12]

Aunque la cabecera no contiene información explícita para distinguir entre una solicitud y una réplica, esta información será vital para comprender estos dos caminos:

- Implícitamente: Por el comando y el contexto en el cual el mensaje es generado.
- Explícitamente: Por el contenido de un paquete encapsulado, en la porción de datos del mensaje.

Tabla 29: LISTA DE COMANDOS

La asignación de códigos de comando será la siguiente:

Código (en hexadecimal)	Nombre	Comentario
0000	NOP	(puede ser usado solamente para TCP)
0001	Reservado por legalidad	
0002 y 0003	Reservado por legalidad	
0004	ListServices	(puede ser usado por

		cualquiera TCP o UDP)
0005	Reservado por legalidad	
0006	Reservado para futura expansión de esta especificación	
0063	ListIdentity	(puede ser usado por cualquiera TCP o UDP)
0064	ListInterfaces	Opcional (puede ser usado por cualquiera TCP o UDP)
0065	RegisterSession	(puede ser usado solamente para TCP)
0066	UnRegisterSession	(puede ser usado solamente para TCP)
00067 hasta el 006F	Reservado por legalidad	
006F	SendRRdata	(puede ser usado solamente para TCP)
0070	SendUniData	(puede ser usado solamente para TCP)
0071	Reservado por legalidad	
0072	IndicateStatus	Opcional (puede ser usado solamente para TCP)
0073	Cancel	Opcional (puede ser usado solamente para TCP)
0074 hasta 00C7	Reservado por legalidad	
00C8 hasta FFFF	Reservado para futura expansión de esta especificación	

Fuente: Traducida de [12]

2.6.2.1.1 Length: (2 bytes) Especifica el tamaño en bytes de la porción de datos del mensaje. Sera 0 para mensajes que no contiene datos. La longitud total de un mensaje será la suma de todos los bytes del paquete de encapsulación.

2.6.2.1.2 Session: (4 bytes) Es generado por el *target* en respuesta al dispositivo *originator*, de una solicitud RegisterSession.

2.6.2.1.3 Status: (4 bytes) Indica si el receptor puede ejecutar comandos de encapsulación. Un valor de 0 en la réplica indica que la ejecución del comando fue realizada. En todos los envíos del emisor, el Status será cero.

Tabla 30 Códigos del atributo Status del objeto identity

CODIGO DE STATUS(EN HEXADECIMAL)	DESCRIPCION
0000	Cumplido exitosamente
0001	El emisor transmitió un comando inválido o que no soporta encapsulación.
0002	Insuficiente recursos de memoria en el receptor para manejar el comando. Esto no es un error de aplicación. En su lugar, esto solamente resulta si la capa de encapsulación no puede obtener recursos memoria que este necesita.
0003	Datos incorrectos, en la porción de encapsulación de un mensaje.
0004-00063	Reservado por legalidad
0064	Un <i>originator</i> uso un manejo de sesión invalido cuando envió el mensaje de encapsulación a la <i>target</i> .
0065	La <i>target</i> recibió un mensaje con longitud invalida.
0066-0068	Reservado por legalidad
0069	No puede soportar la revisión del protocolo
006A-FFFF	Reservado para futura expansión.

Fuente: Traducida de [12]

2.6.2.1.4 Sender context: (8 bytes) Este campo asignara cualquier valor. El receptor retornara este valor sin ninguna modificación en la réplica. Este campo es usado con solicitudes que tengan asociados cierto tipo de réplicas.

2.6.2.1.5 Options: (4 bytes) El emisor colocará un cero. El receptor verificará que en el campo options esté el valor de 0; descartará paquetes que no contengan 0. El propósito de este campo es proveer bits que modifiquen el significado de varios comandos de encapsulación.

2.6.2.2 Descripción de comandos:

2.6.2.2.1 Nop: Un *originator* o un *target* puede enviar un comando NOP. No se genera una réplica para este comando. Sirve para comprobar que una conexión TCP esta mantenida.

2.6.2.2.2 ListIdentity: *Unoriginator* usa este comando para identificar un *target* Ethernet/IP específico, el cual suministre su información de identificación. Este comando será enviado en UDP

2.6.2.2.3 RegisterSession/UnRegisterSession: Estos dos comandos son utilizados para abrir y cerrar una sesión de encapsulación entre dos dispositivos.

Una vez la sesión es establecida, ya se podrán intercambiar mensajes. El dispositivo solicitante crea un valor de Sender Context, y el dispositivo receptor crea una Sesión Handle. Ambos valores son usados para identificar mensajes entre dos dispositivos que usan esta sesión.

Tabla 31: Solicitud RegisterSession

Estructura	Nombre del campo	Tipo de dato	Definición del campo
Cabecera de encapsulación	Command	Uint	RegisterSession (65 hex)
	Lenght	Uint	4 bytes
	Session handle	Udint	0
	Status	Udint	0
	Sender Context	Array de octetos	Cualquier contexto de emisión. Longitud de 8.
	Options	Udint	0
Comando de datos específicos	Protocol version	Uint	La versión del protocolo solicitado será puesto a 1.
	Options flags	Uint	Las opciones de sesión serán puesto a 0. Bits 0-7 son reservados por legalidad. Bits 8-15 son reservados para futuros usos

Fuente: Traducida de [12]

Tabla 32: Replica RegisterSession

Estructura	Nombre del campo	Tipo de dato	Definición del campo
Cabecera de encapsulación	Command	Uint	RegisterSession (65 hex)
	Lenght	Uint	4 bytes
	Session handle	Udint	Valor retornado por el <i>target</i> .
	Status	Udint	0
	Sender Context	Array de octetos	Contexto preservado para la correspondiente solicitud RegisterSession. Longitud de 8
	Options	Udint	0
Comando de datos específicos	Protocol version	Uint	La versión del protocolo solicitado será puesto a 1.
	Options flags	Uint	Las opciones de sesión serán puestos a 0. Bits 0-7 son reservados por legalidad. Bits 8-15 son reservados para futuros usos

Fuente: Traducida de [12]

El comando de la Sesión Handle contiene un identificador generado de *tarjeta* que el *originator* deberá guardar e insertar en el campo de identificador de la sesión handle de todas las solicitudes posteriores a ese *target*.

El campo del Sender Context deberá contener los mismos valores presentes en la solicitud del remitente original. Si el *originator* ha sido registrado con el destino, el status será igual a cero (0). Si el *target* era incapaz de registrar, el status será ajustado a 69 hex (no admitida la revisión del protocolo de encapsulamiento).

El campo de *protocol version* será igual a la versión solicitada si el *originator* se ha registrado correctamente. Si el *target* no soporta la versión solicitada del protocolo, se debe a que:

-La sesión no fue creada.

- El campo *status* deberá establecerse en protocolo de encapsulamiento no compatibles 69 hex.

Si todas opciones solicitadas son compatibles, el campo de *options* devolverá el valor del *originator*. Este valor será igual a cero.

Tabla 33: Comando UnRegisterSession

Un*originator* o *target* puede enviar este comando para terminar la sesión.

Estructura	Nombre del campo	Tipo de dato	Definición del campo
Cabecera de encapsulación	Command	Uint	UnRegisterSession (66 hex)
	Lenght	Uint	0 bytes
	Session handle	Udint	Replica usada para RegisterSession.
	Status	Udint	0
	Sender Context	Array de octetos	Cualquier contexto de envío. Longitud de 8
	Options	Udint	0

Fuente: Traducida de [12]

No hay replica para este comando.

2.6.2.2.4 SendRRData/SendUnitData: El SendRRData es usado para mensajes sin conexión y el SendUniData es usado para mensajes con conexión.

Cuando se encapsula CIP, la solicitud y replica SendRRData son utilizados para encapsular mensajes UCMM.

TABLA 34: Solicitud SendRRData

Estructura	Nombre del campo	Tipo de dato	Definición del campo
Cabecera de encapsulación	Command	Uint	SendRRData(6F hex)
	Lenght	Uint	Longitud de la porción de datos
	Session handle	Udint	Valor usado por el RegisterSession
	Status	Udint	0
	Sender Context	Array de octetos	Cualquier contexto de envío. Longitud de 8
	Options	Udint	0
Comando de datos específicos	Interface Handle	Udint	.será 0 para CIP
	Timeout	Uint	Operación timeout
	Encapsulated packet	Array de octetos	Observar el formato de paquete común sección 2.6.2.3

Fuente: Traducida de [12]

TABLA 35: Replica SendRRData

Estructura	Nombre del campo	Tipo de dato	Definición del campo
Cabecera de encapsulación	Command	Uint	SendRRData(6F hex)
	Lenght	Uint	Longitud de la estructura de datos
	Session handle	Udint	Valor retornado por el RegisterSession
	Status	Udint	0
	Sender Context	Array de octetos	Preservado para la correspondiente solicitud SendRRData
	Options	Udint	0
Comando de datos específicos	Interface Handle	Udint	.será 0 para CIP
	Timeout	Uint	Operación timeout (no se usa)
	Encapsulated	Array de octetos	Observar el

	packet		formato de paquete común sección 2.6.2.3
--	--------	--	--

Fuente: Traducida de [12]

El comando SendUniData enviará mensajes conectados encapsulados. Este comando es usado cuando el protocolo encapsulado tiene su ruta de conexión establecida. No hay réplica para este comando, así que es usado en una comunicación bidireccional T→O y O→T.

TABLA 36: Comando SendUniData

Estructura	Nombre del campo	Tipo de dato	Definición del campo
Cabecera de encapsulación	Command	Uint	SendUniData(70 hex)
	Lenght	Uint	Longitud de la porción de datos
	Session handle	Udint	Valor retornado por el RegisterSession
	Status	Udint	0
	Sender Context	Array de octetos	Cualquier contexto de envió. Longitud de 8.
	Options	Udint	0
Comando de datos específicos	Interface Handle	Udint	será 0 para CIP
	Timeout	Uint	será 0 para CIP
	Encapsulated packet	Array de octetos	Observar el formato de paquete común sección 2.6.2.3

Fuente: Traducida de [12]

2.6.2.3 Administración de una sesión

2.6.2.3.1 Fases para una encapsulación TCP

Una encapsulación debe cumplir 3 fases:

- Establecer una sesión.
- Mantener una sesión.
- Terminar una sesión.

2.6.2.3.2 Establecer una sesión:

Una sesión será establecida apropiadamente de acuerdo a los siguientes pasos:

- El *originator* abre una conexión TCP, usando el puerto AF12 hex.
- El *originator* envía un comando RegisterSession al *target*.
- El *target* chequea la versión del protocolo en el comando del mensaje, para así verificar que soporta el mismo protocolo como el *originator*. Si no es así, la *target* retorna un RegisterSession con un apropiado status.
- La *target* asigna una nueva (única) Sesión ID y envía una réplica del RegisterSession al *originator*.

2.6.2.3.3 Mantener una sesión:

Una vez que la sesión es establecida, se mantendrá hasta que ocurra uno de los siguientes sucesos:

- El *originator* o *target* cierra la conexión TCP.
- El *originator* o *target* envía un comando UnRegisterSession.
- La conexión TCP está quebrada.

2.6.2.3.4 Terminar una sesión:

El *originator* o *target* puede terminar una sesión. La sesión se terminará por cualquiera de estos dos caminos:

-El *originator* o la *target* puede cerrar la conexión TCP. La *target* u *originator* correspondiente detectará la pérdida de la conexión TCP, y cerrará esta parte de la conexión.

-El *originator* o la *target* enviará un comando UnRegisterSession. El envío de un UnRegisterSession detectará la pérdida de la conexión TCP, entonces este cerrará este sitio de la conexión.

2.6.2.4 Formato de paquete común.

El formato de paquete común, es un formato estándar que será empaquetado y transportado, junto con el protocolo encapsulado. El formato de paquete común, es un mecanismo diseñado para acomodar paquetes futuros o tipos de direcciones.

Tabla 37 Formato de paquete común.

NOMBRE DEL CAMPO	TIPO DE DATO	DESCRIPCION
Item de conteo	Uint	Numero de ítems a seguir (al menos serán 2)
Item de direccion	Item Struct (ver más abajo)	Información de dirección para paquetes encapsulados
Item de dato	Item Struct	Los datos de paquete encapsulado

Fuente: Traducida de [12]

La estructura de los ítems data y address será la siguiente:

Tabla 38 Formato ítem de datos y de dirección

Nombre del campo	Tipo de dato	Descripción
Tipo ID	Uint	Tipo de ítem encapsulado
Longitud	Uint	Longitud en bytes del campo dato
Dato	Variable	El dato (si longitud >0)

Fuente: Traducida de [12]

Tabla 39 Números de los ítem id

Numero de Item Id (en hexadecimal)	Tipo de item	Descripcion
0100		Respuesta ListServices
0101-010F		Reservado por legalidad
0110-7FFF		Reservado para futura expansión del protocolo
8000	Dato	Sockaddr Info, originator a target
8001	Dato	Sockaddr Info, target a originator
8002		Item de datos secuenciados
8003-FFFF		Reservado para futura expansión del protocolo
0000	dirección	
0001-000B		Reservado por legalidad
000C		Respuesta ListIdentity

000D-0083		Reservado por legalidad
0084-0090		Reservado para futura expansión del protocolo
0091		Reservado por legalidad
0092-00A0		Reservado para futura expansión del protocolo
00A1	dirección	Basada en conexión (usada para mensajes con conexión)
00A2-00A4		Reservado por legalidad
00A5-00B0		Reservado para futura expansión del protocolo
00B1	Dato	Transporte de paquete de conexión.
00B2	Dato	Mensaje sin conexión.
00B3-00FF		Reservado para futura expansión del protocolo

Fuente: Traducida de [12]

2.6.2.4.1 Ítem de direcciones

2.6.2.4.1.1 Ítem de dirección Nula: El ítem de dirección nula contendrá solamente el tipo ID y el la longitud como se verá a continuación. La longitud será cero. Este será usado cuando el paquete del protocolo no tenga necesidad de enrutar información. Será usado para mensajes sin conexión.

Tabla 40 Ítem de dirección Nula

NOMBRE DEL CAMPO	TIPO DE DATO	Valor del Campo
Tipo ID	UInt	0
Longitud	UInt	0

Fuente: Traducida de [12]

2.6.2.4.1.2 Ítem de direcciónconectada: Este ítem será utilizado cuando el protocolo está orientado a conexión. Los datos contendrán un identificador de conexión.

Tabla 41Ítem de direcciones conectadas

NOMBRE DEL CAMPO	TIPO DE DATO	Valor del Campo
Tipo ID	UInt	A1
Longitud	UInt	4

Dato	Udint	Identificador de conexión
------	-------	---------------------------

Fuente: Traducida de [12]

2.6.2.4.1.3 Ítem de direccionessecuenciadas: Este ítem será usado para transportar paquetes CIP de datos conectados. Los datos contendrán un identificador de conexión y un número de secuencia.

Tabla 42Ítem de dirección secuenciada

NOMBRE DEL CAMPO	TIPO DE DATO	Valor del Campo
Tipo ID	Uint	8002
Longitud	Uint	8
Dato	Udint	Identificador de conexión
	Udint	Numero de secuencia

Fuente: Traducida de [12]

2.6.2.4.2 Ítems de datos

2.6.2.4.2.1 Ítem de datos sin conexión:

El ítem de datos que encapsula un mensaje sin conexión será el siguiente.

Tabla 43 Ítem de datos sin conexión

NOMBRE DEL CAMPO	TIPO DE DATO	Valor del Campo
Tipo ID	Uint	B2
Longitud	Uint	Longitud en bytes, del mensaje sin conexión
Dato	Variable	El mensaje sin conexión.

Fuente: Traducida de [12]

2.6.2.3.2.2 Ítem de datos con conexión:

El ítem de datos que encapsula un mensaje sin conexión será el siguiente:

Tabla 44 Ítem de datos con conexión

NOMBRE DEL CAMPO	TIPO DE DATO	Valor del Campo
Tipo ID	Uint	B1
Longitud	Uint	Longitud en bytes, del paquete transportado

Dato	Variable	El paquete transportado
------	----------	-------------------------

Fuente: Traducida de [12]

2.7 Asignación de la mensajería explícita para TCP/IP

2.7.1 Paquetes CIP sobre TCP/IP:

Cuando el *path* de un paquete CIP, atraviesa una red Ethernet TCP/IP; el paquete encapsulado se transmitirá usando el protocolo TCP/IP y el protocolo encapsulado definido en la sección anterior.

2.7.1.1 Unconnected Messages:

Los paquetes UCMM se transmitirán sobre conexión TCP/IP, usando el protocolo de encapsulación definido en el capítulo anterior. Por ejemplo el formato de la solicitud encapsulada UCMM será como lo muestra la siguiente tabla:

Tabla 45: Solicitud UCMM

ESTRUCTURA	NOMBRE DEL CAMPO		TIPO DE DATO	VALOR DEL CAMPO
Cabecera de encapsulación	Command		Uint	SendRRData (6F)
	Length		Uint	Longitud de la porción de datos
	Session handle		Udint	Valor retornado por el RegisterSession
	Status		Udint	0
	Sender context		Array de 8 octeto	Cualquier sender context
	Options		Udint	0
	Interface handle		Udint	Sera 0 para CIP
Comando de datos específicos	Timeout		Uint	Operación timeout
	Paquete Encapsulado (en el formato de paquete comun)	Item count	Uint	Este campo será 2 desde que un ítem de dirección o ítem de dato sea usado.
		Address Type ID	Uint	Este campo será 0 para indicar un mensaje UCMM.
		Address Length	Uint	Este campo será 0 desde que los mensajes UCMM, use el Item

			de dirección Nula.
		Data Type ID	Uint Este campo sera 00B2 para encapsular el UCMM
		Data Length	Uint Longitud en bytes del siguiente campo (longitud del MR requestpacket)
		MR request packet	Array de Usint Este campo contiene un Formato de solicitud Message Router (tabla 25)

Fuente: Traducida de [12]

Tabla 46: Replica UCMM

ESTRUCTURA	NOMBRE DEL CAMPO		TIPO DE DATO	VALOR DEL CAMPO
Cabecera de encapsulación	Command		Uint	SendRRData (6F)
	Length		Uint	Longitud de la porción de datos
	Session handle		Udint	Valor retornado por el RegisterSession
	Status		Udint	0
	Sender context		Array de 8 octeto	Copia de la correspondiente solicitud UCMM
	Options		Udint	0
	Interface handle		Udint	Sera 0 para CIP
Comando de datos específicos	Timeout		Uint	Operación timeout
	Paquete Encapsulado (en el formato de paquete comun)	Item count	Uint	Este campo será 2 desde que un ítem de dirección o ítem de dato sea usado.
		Address Type ID	Uint	Este campo será 0 para indicar un mensaje UCMM.
		Address Length	Uint	Este campo será 0 desde que los mensajes UCMM,use el Item de dirección Nula.
		Data	Uint	Este campo sera

		Type ID		00B2 para encapsular el UCMM
		Data Length	Uint	Longitud en bytes del siguiente campo (longitud del MR requestpacket)
		MR request packet	Array de Usint	Este campo contiene un Formato de respuesta Message Router (tabla 26)

Fuente: Traducida de [12]

2.7.1.2 Tipos de comunicaciones EtherNet/IP:

Tabla 47 Comunicaciones EtherNet/IP

Tipo de mensaje cip	Relación de comunicación cip	Protocolo de transporte	Tipo de comunicación	Uso típico	Ejemplo
Explícito	Conectados o sin conexión	TCP/IP	Transacciones de solicitud/replica	Información de datos de tiempo no críticos	Parámetros de configuración lectura /escritura
Implícito	Conectados	UDP/IP	Transferencia de datos I/O	Datos I/O de tiempo real	Control de datos de tiempo real para un dispositivo remoto

Fuente: Traducida de [12]

2.7.1.2.1 Mensajería explícita:

En general tiene una naturaleza de petición/respuesta (o cliente/servidor). Este tipo de comunicación se utiliza con datos de tiempo NO real, normalmente para obtener información. Los mensajes explícitos incluyen una descripción de su

significado (expresado explícitamente), por lo que la transmisión es menos eficiente, pero muy flexible.

Puede ser utilizado por un HMI para recopilar los datos, o mediante una herramienta de programación. En los términos CIP, con la mensajería explícita se solicita un servicio de un objeto en particular, por ejemplo, un servicio de lectura o escritura. Para Ethernet/IP, la mensajería explícita utiliza TCP. La mensajería implícita puede realizarse con o sin establecimiento previo de una conexión CIP.

2.7.1.2.2 Mensajería implícita: A menudo se conoce como "I/O" y es de naturaleza de tiempo crítico. Normalmente, este tipo de comunicación se utiliza para el intercambio de datos en tiempo real, donde la velocidad y baja latencia son importantes. Los mensajes implícitos son de muy poca información acerca de su significado, de manera que la transmisión es más eficiente, pero es menos flexible que explícita. La interpretación de los datos transmitidos es rápida. Con mensajes implícitos se puede establecer una asociación (una "conexión CIP") entre dos dispositivos y producir los mensajes implícitos según un determinado mecanismo de activación, normalmente en un determinado índice de paquetes. Ambos dispositivos conocen y están de acuerdo en los formatos de datos que van a utilizar (es decir, el formato es "implícito"). Eficiente, pero menos flexible que el explícito. La interpretación de los datos transmitidos es rápida.

2.7.2 Información sobre el comportamiento del TCP:

TCP es un protocolo fiable orientado a conexión. Si un proceso en cualquiera de los extremos de la conexión se cierra, el TCP en el otro extremo es notificado inmediatamente. Si en un proceso de envío de un mensaje otra no se pueden entregar en un plazo razonable de tiempo, la conexión se supone que se ha roto y se devuelve un error al remitente en todas los posteriores envíos y recibos de la conexión.

Cuando el proceso del *originator* detecta que un destino ha cerrado su extremo de la conexión o que se corta la conexión, se da por supuesto que la conexión se ha interrumpido y cierra su conexión con el destino. Una nueva sesión se establece, como se describe anteriormente a fin de reanudar las comunicaciones con el objetivo.

Aunque un proceso de *originator* recibe una notificación cuando el otro extremo de la conexión se ha cerrado, una ruptura de conexión sólo se puede detectar cuando un proceso realmente intenta enviar un mensaje a través de la conexión. En la mayoría de los casos, el proceso *originator* envía mensajes a objetivos con

suficiente frecuencia de tal forma que un accidente de un equipo de destino se detecta de manera oportuna. Del mismo modo, los objetivos envían mensajes de regreso a los remitentes con mucha frecuencia, para que se detecten rápidamente los procesos de finalización del *originator* y los bloqueos en la máquina. Sin embargo, es posible que un *originator* o *target* no pueda enviar cualquier mensaje en una conexión por un período relativamente largo de tiempo.

El protocolo TCP admite procesos de mantenimiento vivo. Una aplicación puede solicitar TCP para asegurarse de que la conexión sigue trabajando durante los periodos en los que la aplicación no tiene ningún mensaje para enviar. Si esta función está activada, cuando la conexión ha estado inactiva durante cierto período de tiempo, TCP enviará un mensaje de mantenimiento vivo a su par en el otro extremo de la conexión. Si TCP envía varios mensajes de mantenimiento vivo y no recibe una respuesta, TCP supondrá que la conexión se ha roto y se notifica a la solicitud como si hubiera enviado un mensaje que esta fuera de tiempo.

La mayoría de las implementaciones de TCP/IP en procesos de reintentar o de tiempo de espera no declaran un fallo en una conexión, hasta que se ha quedado inutilizable durante varios minutos. Esta es una característica del protocolo TCP en el host de origen; si se reciben mensajes, no los modifica.

2.8 Data Management

La especificación CIP posee una parte denominada Data Management, que puede traducirse al español como la Gestión de los Datos. Ésta describe los modelos de direccionamiento para las entidades CIP y su estructura de datos.

El direccionamiento de la entidad es realizado a través de Segmentos, los que permiten flexibilidad en el uso diferentes tipos de métodos de direccionamiento.

El primer byte de un Segmento CIP permite una distinción entre los siguientes dos esquemas de direccionamiento:

- Segment Address, o Segmento de direccionamiento (00 - 9F hex).
- Data Type, o Tipo de Dato (A0 – DF hex).
- Reservado para usos futuros (E0-FF hex).

Hay varios tipos de segmentos CIP:

- **Port Segment:** Usado para enrutar una subred a otra.

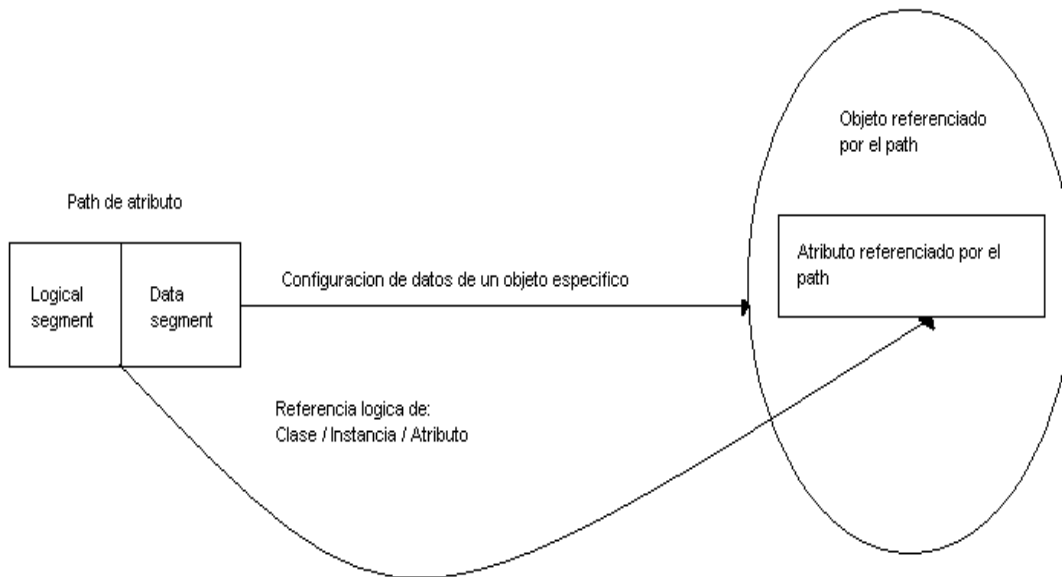
- **Logical Segment:** Referencia información lógica (el cual contiene clase/instancia/atributo)
- **Network Segment:** Especifica parámetros de red que se necesitan en algunas redes.

2.8.1 Uso de los path para segmentos:

El uso de los segmentos CIP sirve para especificar como se van a relacionar objetos diferentes. Un valor usado para especificar una relación, se denomina un path. Un path de atributos consiste en múltiples segmentos, y generalmente referencia la clase, instancia, y el atributo de otro objeto. Un path tiene como tipo de dato un EPATH.

Un ejemplo de path de atributo es mostrado a continuación

FIGURA 7: Ejemplo de path de atributo



Fuente: Traducida de [9]

2.8.1.1 Formato del Path:

Un path (tipo de dato EPATH) puede ser representado en dos formatos diferentes:

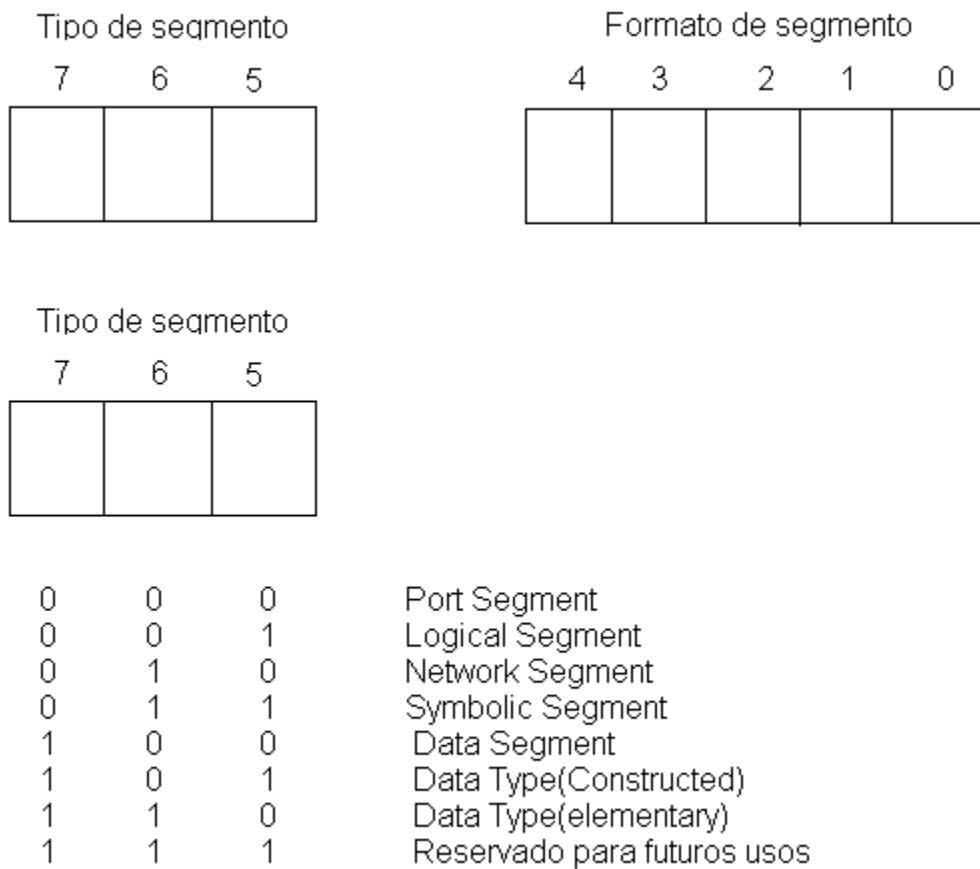
- Padded Path (para indicar que el tipo de datos es Padded EPATH).
- Packet Path (para indicar que el tipo de datos es Packet EPATH).

Cada segmento de un Padded Path será de 16 bits de palabras alineadas. Si un byte pad es requerido para llevar a cabo la alineación, el segmento especificara el lugar del byte pad. Un Packet Path no contiene bytes pad. Cuando un componente es de tipo EPATH, este indicara el formato (Padded o Packet).

2.8.1.2 Tipo de segmento:

Cada segmento de codificación está formado por un tipo de segmento y formato de byte, el cual indica como el segmento será interpretado. La información del segmento tipo/formato está contenida en el primer byte.

FIGURA 8: Codificación del segmento byte tipo/formato



Fuente: Traducida de [9]

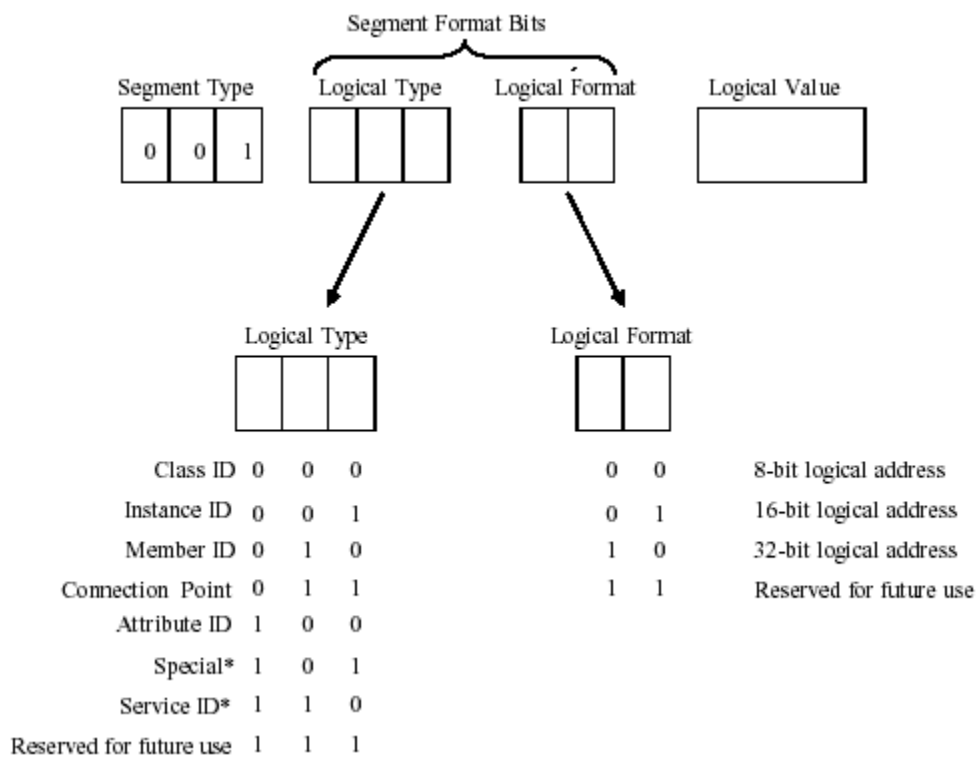
El significado del formato de segmento de bits, dependerá del tipo de segmento.

2.8.1.2.1 Logical segment:

Los logical segment son segmentos de direccionamiento que pueden ser utilizados para conducir un objeto particular dentro de un dispositivo (por ejemplo la clase, atributo e instancia de un objeto). Cuando el logical segment es incluido dentro de un Packet Epath, el valor lógico será anexado en el byte del tipo de segmento.

Cuando el logical segment es incluido dentro de un Padded Epath, el formato de 16 y 32 bits tendrá un pad insertado entre el byte de tipo de segmento y el valor lógico (el formato de 8 bits es idéntico que el Packet Path) El byte pad será colocado como cero.

Figura 9: Codificación del Logical Segment



Fuente: Tomada de [9]

*El tipo de lógica especial e identificadora de servicio no usan formato de dirección lógica.

Los formatos de dirección lógica de 8 bits y de 16 bits, podrán ser usados en todos los tipos de lógica.

El formato de dirección lógica de 32 bit es asignado solamente para la lógica de identificador de instancia y tipo de punto de conexión.

La lógica de punto de conexión provee capacidad de direcciones adicionales más allá del estándar Clase ID/ Instancia ID/ Atributo ID de un objeto. *Una clase de objeto define cuándo y cómo este componente será utilizado.*

El tipo de lógica especial tiene la siguiente definición para el formato de lógica:

- 00 Segmento Electronic key (34 hex).
- 01 Reservado para usos futuros (35 hex).
- 10 Reservado para usos futuros (36 hex).
- 11 Reservado para usos futuros (37 hex).

El segmento Electronic key es usado para la verificación/identificación de un dispositivo. Este segmento tiene el formato que se presenta a continuación.

Tabla 48 Formato segmento Electronic key

NOMBRE DEL CAMPO	TIPO DE DATO	Valor del Campo
Tipo de segmento	Uint	Un valor de 34 hex indica un segmento lógico Electronic Key
Formato Key	Uint	0-3= Reservado 4= Observar la tabla de formato Key
Formato Dato	Array de octeto	Dependerá del formato Key usado.

Fuente: Traducida de [9]

Tabla 49 Formato Key

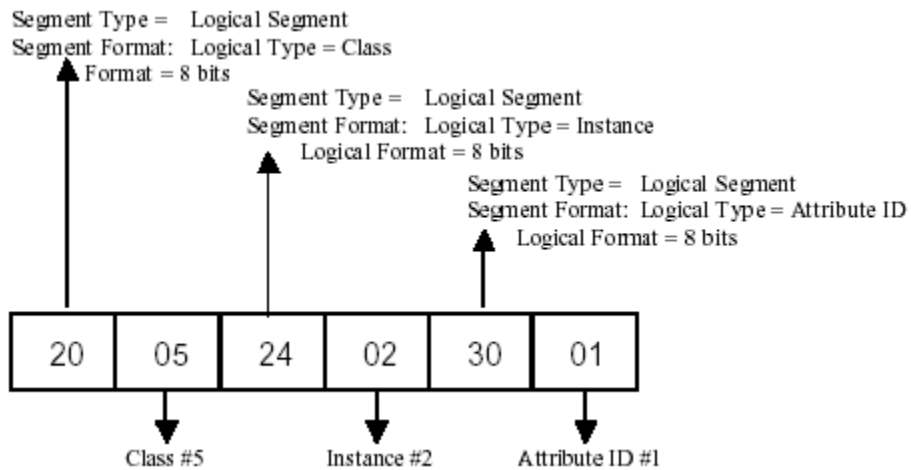
Valor del formato	Valor del campo	Tipo de dato	Semántica
4	Vendor ID	Uint	Vendor ID
	Device type	Uint	Device type
	Product Code	Uint	Product Code
	Major revisión/Compatibilidad	Byte	Bits 0-6=Major revisión Bit 7= Compatibilidad (Si se clarea entonces los atributos sin valor cero entraran en juego. Si se modifica, entonces cualquier key podrá

			ser aceptado como un dispositivo emulado.
	Minor revisión	Usint	Minor revision

Fuente: Traducida de [9]

Una codificación que especifica Clase 5, instancia 2 y atributo 1, es ilustrado a continuación. (Notar que el packet y el padded están igualmente representados).

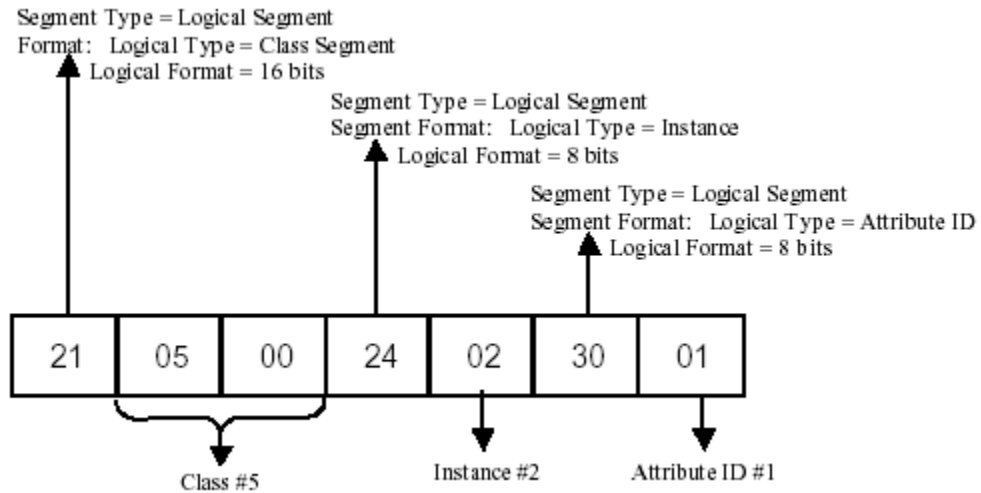
Figura 10 Packet EPATH con 8 bits



Fuente tomada de [10 vol1]

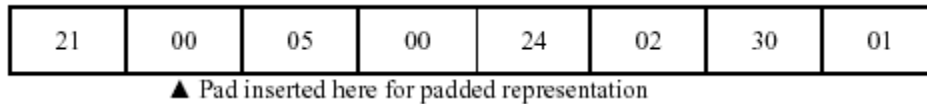
Ahora el mismo ejemplo pero con formato 16 bits. La figura 9 muestra una representación packet y la figura 10 una representación padded.

Figura 11 Packet EPATH con 16 bits



Fuente: Traducida de [9]

Figura 12 Padded EPATH con 16 bits



Fuente: Traducida de [9]

2.8.1.2.2 Data types

Los Data Types, o Tipos de Datos, pueden ser de dos formas:

- Data Type Estructurados.
- Data Type Elementales.

Los Data Type Estructurados pueden ser arreglos de Data Type Elementales o cualquier composición de arreglos de Data Type Elementales.

2.8.1.2.2.1 Data types Elementales:

Los data type Elementales y los valores (rangos) de las variables de cada tipo son mostrados en la siguiente tabla:

Tabla 50 Data types elementales

Tipo de dato	Descripcion	Rango	
		Minimo	Maximo
BOOL Boolean	Booleano	Nota 1	

SINT	Entero corto	-128	127
INT	Entero	-32768	32767
DINT	Doble entero	-2^{31}	$2^{31}-1$
LINT	Entero largo	-2^{63}	$-2^{63}-1$
USINT	Entero corto sin signo	0	255
UINT	Entero sin signo	0	65535
UDINT	Entero doble sin signo	0	$2^{32}-1$
ULINT	Entero largo sin signo	0	$2^{64}-1$
REAL	Punto flotante	Nota 2	
LREAL	Flotante largo	Nota 3	
ITIME	Duracion(corta)	Nota 12	
TIME	Duracion	Nota 4	
FTIME	Duracion (alta resolucion)	Nota 5,6	
LTIME	Duracion (larga)	Nota 6,7	
DATE	Dato solamente	Nota 8	
TIME_OF_DAY o TOD	Tiempo del dia	Nota 9	
DATE_AND_TIME o DT Date	Dato y tiempo del dia	Nota 10	
STRING	Caracter string (1 byte por caracter)		
STRING2	Caracter string (2 bytes por caracter)	Nota 6	
STRINGN	Caracter N (N bytes por caracter)	Nota 6	
SHORT_STRING	Caracter string (1 byte por caracter,1 indicador de longitud de byte)	Nota 6	
STRINGI	Caracter string internacional	Nota 6	
BYTE	Bit string-8 bits	Nota 11	
WORD	Bit string-16 bits	Nota 11	
DWORD	Bit string-32 bits	Nota 11	
LWORD	Bit string-64 bits	Nota 11	
EPATH CIP	Segmento path	Nota 13	

Fuente: Traducida de [9]

1: Los valores de tipo BOOL 0 y 1, corresponden a las palabras clave falso o verdadero.

2: El rango de valores de tipo real están definidos en la IEEE754 para el formato de un solo punto flotante.

3:El rango de valores de tipo reales están definidos en la IEEE754 para el formato doble de punto flotante.

4: El rango de valores de tipo TIME es la misma que la de las variables de tipo DINT, que representa el tiempo en milisegundos, es decir, un rango de T# -24d20h31m23.648s hasta T# 24d20h31m23.647s.

5:El rango de valores de tipo FTIME es la misma que la de las variables de tipo DINT, que representa el tiempo en milisegundos, es decir, un rango de T#-35m47.483648s hasta T#35m47.483547s.

6:Este es un estándar CIP extendido de la IEC 1131-3.

7: El rango de valores de tipo LTIME es la misma que la de las variables de tipo LINT, que representa el tiempo en milisegundos, es decir, un rango de T#-106751991d4h0m54.775808s hasta T#106751991d4h0m54.775807s.

8:El rango de valores de las variables de tipo DATE es del D# 1972-01-01, el inicio de la hora universal coordinada con sus siglas (UTC) era, hasta D# 2151-06-06 (un total de 65.536 días).

9:El rango de valores de las variables de tipo TIME_OF_DAY es de TOD#00:00:00,000 hasta TOD#23:59:59,999 para una resolución de 1 milisegundo.

10:El rango de valores de las variables de tipo DATE_AND_TIME es desde DT#1972-01-01-00:00:00.000 hasta DT#2151-06-06-23:59:59.999.

11: Los valores para los tipos de datos bit string está en el rango de 2#bN-1bN...b2b1b0, donde N es el número de bits en el bit string, bN-1 es el “bit más significativo”, y b0 es el “bit menos significativo”. El valor del J-ésimo bit es representado como 0 o 1, que corresponde al valor Boleano FALSO o VERDADERO, respectivamente.

12: El rango de valores de tipo ITIME es la misma que la de las variables de tipo INT, que representa el tiempo en milisegundos, es decir, un rango de T#-32s768ms hasta T#32s767ms.

13: Ver la sección 2.7.1.

2.9 EDS ELECTRONIC DATA SHEETS

El protocolo CIP entrega varias opciones para la configuración de dispositivos remotos a través de la red y parámetros embebidos en estos. Usando estas características, se podrá seleccionar y modificar una configuración para una aplicación particular.

Los métodos de configuración son los siguientes:

- Por hojas de datos impresa.
- Un EDS Electrónico Data Sheet.
- Una configuración ensamblada de los métodos de arriba.
- Un device type manager o administrador de perfil de dispositivo (DTM).

De todas las alternativas, la más eficiente es el uso de Electronic Data Sheet o EDS.

Un EDS es un archivo de texto escrito en ASCII y que se puede generar en cualquier editor ASCII (por ejemplo, el bloc de notas de Windows). La especificación CIP proporciona un completo sistema de reglas para el diseño y la sintaxis de un EDS, lo cual hace que la configuración de los dispositivos sea más simple. El propósito principal de un EDS es entregar la información sobre varios aspectos de las capacidades del dispositivo, siendo las más importantes las conexiones I/O que soporta y qué parámetros de configuración existen dentro del dispositivo.

El EDS provee de información del producto manufacturado al usuario del producto. El usuario del producto interpreta el data sheet manufacturado, decide que modificaciones se deben hacer para valores que no están por defecto y ejecuta acciones necesarias para obtener la información de los datos dentro del dispositivo.

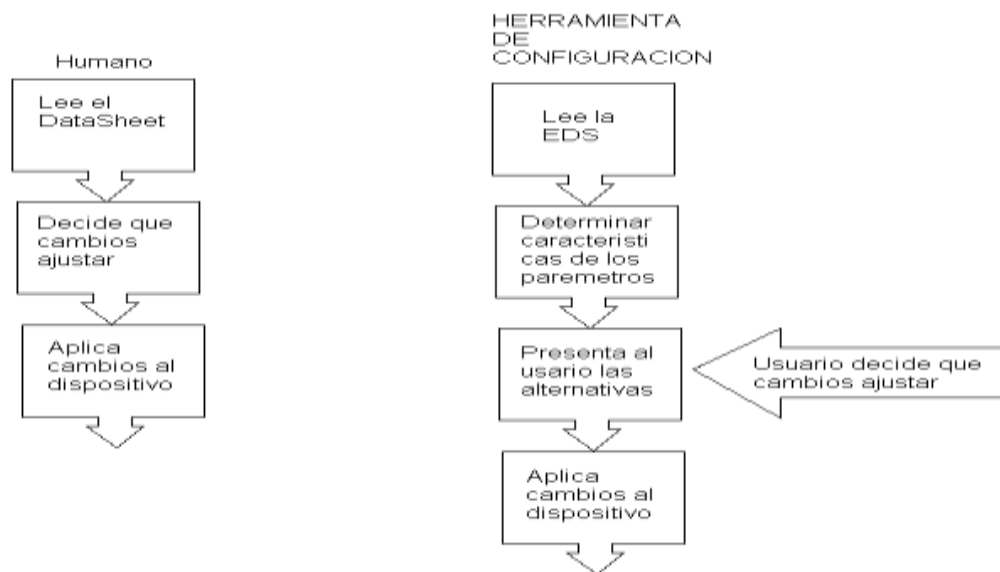
El servidor EDS tiene dos grandes propósitos especiales:

- Describe cada parámetro del dispositivo, incluyendo la legalidad de este y valores por defecto.
- Provee al usuario una selección de opciones para cada parámetro configurable de un dispositivo.

La figura que se muestra a continuación, compara la interpretación de una hoja de datos impresa hecha por un humano y un EDS como una herramienta de configuración. Como mínimo, una herramienta de configuración CIP debe tener:

- Cargar la EDS dentro de la memoria de herramienta de configuración.
- Interpretar el contenido de la EDS para determinar las características de cada parámetro.
- Presentar al usuario una lista de opciones o de datos de entrada para los campos de los parámetros de cada dispositivo.
- Cargar la lista de opciones de parámetros para usuario, a los parámetros correctos en el dispositivo.

FIGURA 13: Ilustración del uso de una hoja datos impresa vs EDS



Fuente: Traducida de [9]

La especificación EDS suministra una consistente y apropiada compatibilidad para un buen desempeño en el ambiente CIP. Esta especificación hace cumplir los requerimientos críticos para llevar a cabo:

- Consistencia entre vendedores para el beneficio de la construcción de dispositivos.

-Compatibilidad entre vendedores para el beneficio de la construcción de herramientas.

Todos los desarrolladores implementaran EDS compatibles con esos requerimientos. Los productos desarrollados determinan todos los otros detalles de implementación. Cada intérprete de un EDS diseñado para productos CIP, estará en la capacidad de:

- Leer e interpretar cualquier estándar EDS.
- Presentar información y selección de los usos del dispositivo.
- Construir los mensajes necesarios para configurar los productos CIP asociados.

2.9.1 Sección de un EDS:

Un EDS se estructura en secciones. Cada sección comienza con un nombre de sección entre corchetes []. Las primeras dos secciones son obligatorias para todo el EDS (colocaremos los que pueden ser necesarios para EtherNet/IP):

- [File]: Describe el contenido y la revisión del archivo.
- [Device]: Es equivalente a la información del Identity Object y se utiliza para emparejar un EDS a un dispositivo.
 - [Device Classification]: Describe con qué red se puede conectar el dispositivo.
 - [ParamClass]: Describe atributos del nivel de clase del Parameter Object.
 - [Params]: Identifica todos los parámetros de configuración en el dispositivo, sigue la definición del Parameter Object.
 - [EnumPar]: Lista de enumeración de opciones de parámetro para presentar al usuario.
 - [Connection Manager]: Describe las conexiones soportadas por el dispositivo.
 - [Port]: Describe los varios puertos de red que un dispositivo puede tener.
 - [Modular]: Describe las estructuras modulares dentro de un dispositivo.

– [Capacity]: Especifica la capacidad de comunicación del dispositivo EtherNet/IP.

De todas estas secciones las requeridas en todo EDS son la sección File y la sección Device. Estas serán descritas con más profundidad en la sección 2.8.3.

Nota: Un EDS para un dispositivo EtherNet/IP no se refiere directamente al enlace del objeto para el puerto EtherNet/IP, (por ejemplo, el objeto Ethernet link) desde que este pueda ser referenciado a través del objeto TCP.

FIGURA 14: Ejemplo de EDS para un dispositivo EtherNet/IP

```
[File]
  DescText = "Widget EDS File";
  CreateData = 02-07-2001;
  CreateTime = 17:51:44;
  ModDate = 04-06-1997;
  ModTime = 22:07:30;
  Revision = 2.1;
  HomeURL = "http://www.controlnet.org/EDS/12345.eds";
[Device]
  VendCode = 65535;
  VendName = "Widget-Works, Inc.";
  ProdType = 0;
  ProdTypeStr = "Generic";
  ProdCode = 10;
  MajRev = 1;
  MinRev = 1;
  ProdName = "Smart-Widget";
  Catalog = "1492-SW";
  Icon = "widget.ico";
[Device Classification]
  Class1 = EthernetIP;
[Port]
  Port1 =
    TCP,
    "EtherNet/IP port",
    "20 F5 24 01",
    1;
```

Fuente: Tomada de [12]

2.9.2 Reglas de diseño de un EDS:

Observar estas reglas cuando se define una EDS:

-EDS Espacio en blanco: El intérprete del EDS tendrá en cuenta caracteres como caracteres en espacio en blanco. Esos caracteres de espacio en blanco, son significativos para el intérprete. Estos son:

-Nueva línea.

-Carriage return: Tecla idéntica a la tecla ENTER

- Linefeed:Es un código que mueve al cursor en una pantalla una línea para abajo; en una impresora, hace avanzar el papel una línea.

- Tabulaciones, vertical y horizontal.

- Formato de relleno.

- Marcador de fin de archivo.

- Comentarios.

- Caracteres de palabras clave: Todas las palabras claves dentro de un archivo EDS, deberán estar compuesto por caracteres ASCII según la siguiente lista:

- Letras mayúsculas de la A a la Z.

- Letras minúsculas de la A a la Z.

- Números del 0 al 9.

- El caracter especial para subrayar “_”

- El caracter espacio. (El espacio puede ser usado solamente en una sección de palabras claves. El espacio puede aparecer solamente dentro de la sección de un nombre, y múltiples secuencias de espacios son inválidas).

- Secciones –Los archivos EDS deberán estar clasificados en secciones requeridas y opcionales.

- Delimitaciones de secciones - Cada sección de una EDS deberá estar propiamente delimitada para una *sección de palabras clave*.

- Secciones de palabras clave – Cada sección de palabras clave está definido para ser el texto entre el inicio de una de sección palabras clave delimitadas. Los caracteres validos que son usados en la sección palabras clave son mostrados arriba. Hay dos tipos de sección de palabras clave: públicas y de vendedor específico.
- Orden de sección: Cada sección requerida deberá ser colocada en el orden de requerimiento.
- Entrada – Cada sección de un EDS contiene una o más *entradas* de inicio, con una palabra clave de entrada seguida por un signo igual. El significado de la palabra clave de entrada depende del contexto de la sección. Un punto y coma indica el fin de la entrada, y las entradas pueden expandir múltiples líneas.
- Entrada de palabras clave – Una entrada de palabras clave consiste en una única secuencia de caracteres de palabras clave. Hay dos tipos de entrada de palabras clave, público y de vendedor específico.
- Palabra clave pública – Una palabra clave pública será definida por la ODVA. Una palabra clave pública nunca iniciara con cualquier dígito numérico.
- Campos de entrada – Cada entrada contiene uno o más campos. Una coma separa cada campo.
- Un espacio en blanco o nada entre comas (por ejemplo: palabra clave=x,,x;).
- Un espacio en blanco o nada entre una coma y un punto y coma (por ejemplo =x,;).
- Un espacio en blanco o nada entre un igual y coma (por ejemplo: palabra clave=,x;).
- Un espacio en blanco o nada entre un igual y un punto y coma (por ejemplo: palabra clave=;).
- Palabras clave de vendedor específico –La sección y las entradas de palabras clave pueden ser de vendedor específico. Las secciones comienzan con la ID de vendedor de la compañía seguida de un guion bajo (ID de vendedor_palabra clave de vendedor específico). Las entradas de palabras clave de un vendedor específico en una sección de vendedor específico serán siempre propias del vendedor; ellas no requieren adicionar la ID de vendedor de la compañía

fabricante. La Id de vendedor será entregada en decimal y no contendrán ceros de cabecera. Cada vendedor es responsable para mantener y documentar esas palabras clave de vendedor específico.

2.9.3 Descripción de las secciones de un EDS

2.9.3.1 Descripción de la sección File: La sección file comienza con la palabra clave [File] y contiene información acerca del archivo EDS. Una herramienta de configuración lee esta información, formatea este, y se muestra al usuario. El usuario también puede acceder a esta sección con un visor de archivos de texto sin formato y mostrar la información. Esta sección no requiere modificación alguna a menos que el usuario modifique el archivo manualmente.

La descripción de la sección File debe contener:

Tabla 51: Campos de la sección File

NOMBRE DE ENTRADA	PALABRA CLAVE	TIPO DE DATO	REQUERIDO /OPCIONAL
File Description Text	DescText	ASCCI Character Array	Requerido
File Creation Date	CreateDate	DATE	Requerido
File Creation Time	CreateTime	TIME_OF_TIME	Requerido
Last Modification Time	ModTime	TIME_OF_DAY	Condicional
Last Modification Date	ModDate	REVISION	Condicional
Eds Revision	ModTime	ASCCI Character Array	Requerido
Home Url	Revision	ASCCI Character Array	Opcional

Fuente: Traducida de [9]

2.9.3.1.1 File description text: Es una simple línea de texto que se muestra en la herramienta de configuración. El desarrollador de la EDS asigna un significado para esta línea de texto. Todos los caracteres están encerrados entre comillas.

2.9.3.1.2 File Creation Date: Es la fecha de creación del EDS. Asignado por el EDS. Sólo para mayor conveniencia, se puede utilizar esta fecha para obtener información de la versión del archivo. Una herramienta de configuración no utiliza esta información para llevar a cabo cualquier tipo de versión de control, pero puede mostrar el contenido.

2.9.3.1.3 File Creation Time: Es la hora de creación del EDS. Sólo para mayor conveniencia, se puede utilizar este dato para obtener información de la versión del archivo. Una herramienta de configuración no utiliza esta información para llevar a cabo cualquier tipo de versión de control, pero puede mostrar el contenido.

2.9.3.1.4 Last modification Date: Es la fecha de la última modificación a la EDS. Una herramienta de configuración que permita modificar el archivo EDS, deberá actualizar este campo si es necesario. Una herramienta de configuración muestra el contenido de esta entrada, si puede cambiar la EDS, y por lo tanto actualizar este campo.

Además se necesitara si la palabra clave de la *Last Modification Time* está presente.

2.9.3.1.5 Last modification Time: Es la hora de la última modificación a la EDS. Una herramienta de configuración que permita modificar el archivo EDS, deberá actualizar este campo si es necesario. Una herramienta de configuración muestra el contenido de esta entrada, si puede cambiar la EDS, y por lo tanto actualizar este campo.

2.9.3.1.6 Eds revision: Es la revision del EDS. La revision de la EDS no es requerida para tener una relación con la revisión del producto, esta es simplemente la revisión misma del archivo EDS. La revisión puede tener un formato así:

Major_revision.minor_revision

Por ejemplo: 2.1 <= major revisión es 2, y minor revisión es 1.

2.9.3.1.7 Home URL: Es un localizador uniforme de recursos del archivo maestro de la EDS, el icono archivo y otros archivos descritos de la EDS. La URL de inicio

deberá especificar una URL argumentada para referenciar una versión maestra del archivo EDS.

2.9.3.2 Descripción de la sección Device:

La descripción de la sección device comienza con la palabra clave [Device] y contiene información de la manufactura del dispositivo, incluye alguno de los valores del objeto identity. La tabla que se muestra a continuación muestra las entradas de la sección device.

Tabla 52: Campos de la sección Device

NOMBRE DE ENTRADA	PALABRA CLAVE	TIPO DE DATO	REQUERIDO/OPCIONAL
Vendor ID ^{1,2}	VendCode	Uint	Requerido
Vendor name	VendName	ASCCI Character Array	Requerido
Device type ^{1,2}	ProdType	Uint	Requerido
Device Type String	ProdTypeStr	ASCCI Character Array	Requerido
Product Code ^{1,2}	ProdCode	Uint	Requerido
Major revision ^{1,2}	MajRev	Usint	Requerido
Minor revision ¹	MinRev	Usint	Requerido
Product name ¹	ProdName	ASCCI Character Array	Requerido
Catalog number	Catalog	ASCCI Character Array	Opcional
Icon file name	Icon	ASCCI Character Array	Opcional

Fuente: Traducida de [9]

1. Esta entrada representa un atributo del objeto identity.
2. Esta entrada es usada para confrontar un EDS con una revisión específica de un producto.

La descripción de los atributos del objeto identity están descritos en la sección 2.5.1.1

2.9.3.2.1 Catalog Number: Es el nombre del catálogo o número de modelo. Uno o más número de catálogos podrán estar asociados con un código de producto particular. En el caso de varios números de catálogo, es útil proporcionar un número de catálogo para ser más práctico. Por ejemplo, 1438- BAC7xx donde “xx” representa variantes en el número de catálogo para el código del producto.

2.9.3.2.2 Icon File Name: Nombre de archivo o icono de archivo. Identifica un archivo que contiene una representación gráfica del archivo. El archivo tendrá el formato WINDOWS ICON (ICO) [13], y contiene como mínimo 16x16 iconos. El archivo puede contener 32x32, 48x48, y 64x64 iconos. La ubicación del archivo Icon es la combinación de la ubicación especificada por la HomeURL y el nombre de archivo especificado en esta palabra clave. Esta palabra clave está presente cuando una palabra clave HomeURL exista.

2.9.3.3 Descripción de la sección Device Classification:

La sección de clasificación de dispositivo comienza con la palabra clave [Device Classification] y clasificará el dispositivo descrito por la EDS dentro de una o más categorías. La entrada de la palabra clave para todas las clasificaciones consistirá en el array carácter, “Class”, combinada con un número de decimal. Los números comienzan en 1 para la primera clase, y se incrementara por cada clase adicional. Las comas separaran todos los campos, y un punto y coma indicara el fin de la entrada.

El número de campos para cada entrada de clasificación será variable para permitir una clasificación de forma de árbol, similar a los que utilizan los sistemas de archivos en una estructura de directorios.

Las subclasificación de las clasificaciones públicas serán reservadas. La clasificación de un vendedor específico puede ser subclasificado a discreción del vendedor.

El primer campo representa el mayor nivel en una estructura de árbol y será unos de los siguientes:

- CompoNet;

- ControlNet;
- DeviceNet;

- EtherNet/IP

- ModbusSL

- ModbusTCP

- Un campo de vendedor específico.

El campo de vendedor específico comenzara con la identificación de vendedor de la compañía fabricante seguida de un guión bajo (VendorID_VendorSpecificField). La ID de vendedor será entregada en decimal y no tendrá ceros de cabecera. Cada fabricante es responsable de mantener y documentar el campo de vendedor específico.

2.9.3.4 Descripción de la sección Port:

La sección Port comienza con la palabra clave [Port] y describe los puertos CIP dentro de un dispositivo. Cada CIP tendrá una entrada correspondiente en esta sección. La palabra clave de entrada consiste en el array carácter “Port”, combinada con un número decimal correspondiente a una instancia del objeto puerto. Por ejemplo, Port1 es la instancia 1 del objeto Port. Un espacio en blanco o nada entre comas será usado para campos opcionales no previstos.

La entrada del puerto N corresponde al puerto compatible EtherNet/IP que puede ser asignado de la siguiente forma:

El “Port Type” tendrá un valor de “TCP”

El campo opcional “Port Object” será fijado en la ruta del objeto TCP para este puerto.

FIGURA 15: Ejemplo de la sección Port

```
[Port]
Port1 = DeviceNet,
"Port A",           $ name of port
"20 03 24 01",     $ instance one of the DeviceNet object
2;                 $ port number 2

Port2 = 65535_Chassis,
"Chassis",         $ name of port
"20 9A 24 01",     $ vendor specific back-plane object
1;                 $ port number 1
```

Fuente: Tomada de [9]

2.10 CODIGOS DE ESTADOS GENERAL

En la siguiente tabla se enlistan los códigos de estados, que pueden estar presentes en el campo de códigos de estado general en un mensaje con respuesta errada.

Tabla 53: CODIGOS DE ESTADOS GENERAL CIP.

CODIGO GENERAL DE ESTADO (en hexadecimal)	NOMBRE DEL ESTADO	DESCRIPCION DEL ESTADO
00	Realizado	El servicio se llevó a cabo exitosamente para el objeto especificado.
01	Falla de conexión	Un servicio relacionado con la falla de conexión a lo largo de la ruta de conexión.
02	Recurso no disponible	El servicio no suministro un atributo en una lista de atributos, que era necesario para el servicio, para hacer el procedimiento de la solicitud.
03	Valor de parámetro invalido	Observar el condigo de estado 20, que define el valor usado adecuadamente para esta

		condición
04	Error de segmento path	El path segment identificado o la sintaxis del path no fue comprendido por el nodo de proceso. El proceso path, deberá detenerse cuando un hay un error de path segment.
05	Path de destino desconocido	El path segment está referenciando a una clase de objeto, instancia que no es conocido o no se contiene en el nodo de proceso. El proceso path, deberá detenerse cuando un hay un error de path segment.
06	Transferencia parcial	Sólo una parte de los datos esperados fue transferida.
07	Conexión perdida	La mensajería ha perdido la conexión.
08	Servicio no soportado	El servicio solicitado no fue implementado o no estaba definido para esta clase de objeto o instancia.
09	Valor de atributo invalido	Datos de atributos inválidos detectado.
0A	Error en la lista de atributos	Un atributo en el Get_Attribute_List o Set_Attribute_List respuesta tiene un estado distinto de cero.
0B	Ya está en modo de solicitud	El objeto ya está en el modo o estado solicitado por el servicio
0C	El objeto en estado de conflicto	El objeto no puede realizar el servicio solicitado en su actual estado o modo.
0D	El objeto ya existe	La instancia de objeto que se va a crear ya

		existe.
0E	Atributo no configurable	Una solicitud para modificar un atributo no modificable se recibió.
0F	Violación privilegiada	Comprobar falló en un permiso/privilegio
10	Dispositivo en estado de conflicto	El dispositivo en su actual modo o estado prohíbe la ejecución del servicio solicitado
11	Datos de respuesta demasiado grande	La transmisión de los datos en el búfer de respuesta es mayor que el búfer respuesta asignado
12	Fragmentación de un valor primitivo	El servicio especifica una operación que va a fragmentar un valor primitivo de datos, es decir, la mitad de los datos de tipo REAL.
13	No hay datos suficientes	El servicio no proporciona datos suficientes para realizar la operación especificada
14	Atributo no soportado	El atributo especificado en la solicitud no es compatible
15	Demasiados datos	El servicio suministra más datos de los que se esperaba
16	El objeto no existe	El objeto especificado no existe en el dispositivo
17	El Servicio de secuencia de fragmentación no está en curso	La fragmentación secuenciada para este servicio, no está activada para estos datos.
18	No almacena datos de atributos	Los datos de atributos de este objeto no se ha guardado con anterioridad al servicio solicitado.
19	Fracaso de operación de	Los datos de atributos

	almacenamiento	de este objeto no se ha guardado debido a un fallo durante el intento
1A	Fracaso enrutamiento, paquete de solicitud demasiado grande	El paquete de servicio de solicitud era demasiado grande para la transmisión en una red en la ruta hacia el destino. El dispositivo de enrutamiento se vio obligado a interrumpir el servicio.
1B	Fracaso enrutamiento, paquete de respuesta demasiado grande	El paquete de servicio de respuesta era demasiado grande para la transmisión en una red en la ruta hacia el destino. El dispositivo de enrutamiento se vio obligado a interrumpir el servicio.
1C	Faltan datos en la lista de atributos de entrada	El servicio no suministro un atributo en una lista de atributos, que necesita el servicio para hacer la característica de la solicitud.
1D	Atributo invalido en la lista de valores	El servicio está devolviendo la lista de atributo, suministrando información de estado para aquellos atributos que son válidos.
1E	Error de servicio embebido.	Un servicio embebido resulta en un error.
1F	Error de proveedor específico.	Un error de un proveedor específico ha sido encontrado. El Uso General de este código de error, sólo debe realizarse cuando no hay códigos de error presentados en esta tabla o dentro de una clase de objeto que

		defina con exactitud el error
20	Parámetro invalido	Un parámetro relacionado con la solicitud no es válido. Este código se utiliza cuando un parámetro no cumple con los requisitos de la presente especificación y/o los requisitos definidos en un objeto de aplicación específica.
21	Una sola escritura o ya ha sido escrito.	Se hizo un intento de escribir en una medio de escritura-una vez(p.ej. Unidad WORM, PROM) que ya se ha escrito, o para modificar un valor que no se puede cambiar una vez establecido.
22	No válida respuesta recibida	Una respuesta no válida es recibida (p. ej., la respuesta de código de servicio no coincide con el código de servicio de la petición, o el mensaje de respuesta es menor que la mínimo tamaño de respuesta esperado). Este código de estado puede servir para otras causas de respuestas no válidas.
23	Desbordamiento de búfer	El mensaje recibido es más grande que lo que búfer puede recibir para manipular. Todo el mensaje fue descartado.
24	Formato de mensaje Error	El formato del mensaje recibido no es compatible con el servidor.
25	Falla del key en el path	El segmento key que se incluyó en el primer segmento del key no coincide con el módulo

		destino. El estado del objeto específico indicará que parte de la verificación del key ha fallado.
26	Tamaño del path invalido	El tamaño del path que se ha enviado con la petición de servicio no es lo suficientemente grande como para permitir que la solicitud se dirija a un objeto o demasiado enrutamiento de datos se incluyó.
27	Atributo inesperado en la lista	Se hizo un intento por definir un atributo que no es capaz de establecerse en este momento
28	Invalido Miembro ID	El miembro ID especificado en la solicitud no existe en la Clase/Instancia/Atributo.
29	Miembro no configurable	Una solicitud para modificar un miembro no modificable fue recibido
2C-CF		Reservado por CIP para futuras expansiones

Fuente: Traducida de [9]

3. LOS SISTEMAS EMBEBIDOS

3.1 Definición para sistema embebido:

Un sistema embebido o sistema empotrado se define como un sistema electrónico diseñado específicamente para realizar unas determinadas funciones, habitualmente formando parte de un sistema de mayor entidad. La característica principal es que emplea para ello uno o varios procesadores digitales (CPUs) en formato microprocesador, microcontrolador lo que le permite aportar 'inteligencia' al sistema anfitrión al que ayuda a gobernar y del que forma parte.

En este conjunto nos podemos fácilmente imaginar la necesidad un circuito electrónico que contenga los diferentes programas de lavado del que disponga el electrodoméstico. Y por razones obvias, esta electrónica no sería otra cosa que un

micro-ordenador, especialmente diseñado para dicho fin. Por lo tanto podemos hablar de un sistema embebido o empotrado en el interior de la lavadora [14].

Se entiende por sistemas embebidos a una combinación de hardware y software de computadora, sumado tal vez a algunas piezas mecánicas o de otro tipo, diseñado para tener una función específica. Es común el uso de estos dispositivos pero pocos se dan cuenta que hay un procesador y un programa ejecutándose que les permite funcionar.

Esto ofrece un contraste con la computadora personal, que también está formada por una combinación de hardware y software más algunas piezas mecánicas (discos rígidos, por ejemplo). Sin embargo la computadora personal no es diseñada para un uso específico. Si no que es posible darle muchos usos diferentes.

Muchas veces un sistema embebido es un componente de un sistema mucho más grande, como por ejemplo los sistemas de frenos o el sistema de inyección de combustible, en automóviles actuales son sistemas embebidos.

Esta combinación de software y hardware puede ser reemplazada en muchos casos por un circuito integrado que realice la misma tarea. Pero una de las ventajas de los sistemas embebidos es su flexibilidad, ya que a la hora de realizar alguna modificación resulta mucho más sencillo modificar unas líneas de código al software del sistema embebido, que reemplazar todo el circuito integrado.

Un uso muy común de los sistemas embebidos es en los sistemas de tiempo real, entendiéndose por sistemas en tiempo real a aquellos sistemas en los que el control del tiempo es vital para el correcto funcionamiento. Los sistemas en tiempo real necesitan realizar ciertas operaciones o cálculos en un límite de tiempo. Donde ese límite de tiempo resulta crucial [15].

3.2 Componentes de un sistema embebido:

Un sistema embebido en principio estaría formado por un microprocesador y un software que se ejecute sobre éste. Sin embargo este software necesitara sin duda un lugar donde poder guardarse para luego ser ejecutado por el procesador.

Esto podría tomar la forma de memoria RAM o ROM, Todo sistema embebido necesitara en alguna medida una cierta cantidad de memoria, la cual puede incluso encontrarse dentro del mismo chip del procesador. Además de esto normalmente un sistema embebido contara con una serie de salidas y entradas necesarias para comunicarse con el mundo exterior.

Debido a que las tareas realizadas por sistemas embebidos son de relativa sencillez, los procesadores comúnmente usados cuentan con registros de 8 o 16 bits.

En su memoria sólo reside el programa destinado a gobernar una aplicación determinada. Sus líneas de entrada/salida soportan el conexionado de los sensores y actuadores del dispositivo a controlar y todos los recursos complementarios disponibles tiene como única finalidad atender a sus requerimientos.

Estas son las únicas características que tienen en común los sistemas embebidos, todo lo demás será totalmente diferente para cada sistema embebido en particular debido a la inmensa diversidad de aplicaciones disponibles.

3.3 Sistemas embebidos en Ethernet:

La Ethernet conmutada permite a los usuarios la transición de la mensajería estadística a la conmutación basada en hardware, totalmente determinista. La tendencia es «aplanar» la estructura de la red y utilizar el menor número posible de tecnologías para reducir el coste de propiedad.

Ethernet se está convirtiendo en la red universal a todos los niveles de la empresa. Actualmente hay instalaciones de amplio campo de aplicación con las correctas herramientas y centenares de nodos PC de sobremesa: todas las herramientas, capacidades y recursos necesarios para mantener operativa una gran red comercial 24 horas al día, siete días a la semana. La preferencia por Ethernet en las instalaciones de sobremesa se debe al bajo costo de su instalación y mantenimiento y su capacidad de integración con Internet. Pero existe una tendencia paralela y de rápido movimiento en la automatización industrial y, si los entusiastas de las redes domésticas tienen esta posibilidad, también existe en el hogar. La inteligencia se está desplazando al nivel más bajo de la red, no sólo para datos, sino también para control. El ordenador central que lo controla todo es ya un concepto obsoleto.

En cambio la idea de que un dispositivo puede controlarse a sí mismo gana cada vez más importancia. Todo esto apunta a la nueva generación de redes que descienden directamente a los dispositivos embebidos alimentados por la tecnología del chip en el propio sistema. Ethernet puede convertirse en la red dominante para la interconexión de dispositivos embebidos, como ya lo es en instalaciones de sobremesa.

Depende de la importancia de la inversión que desee hacer un usuario.

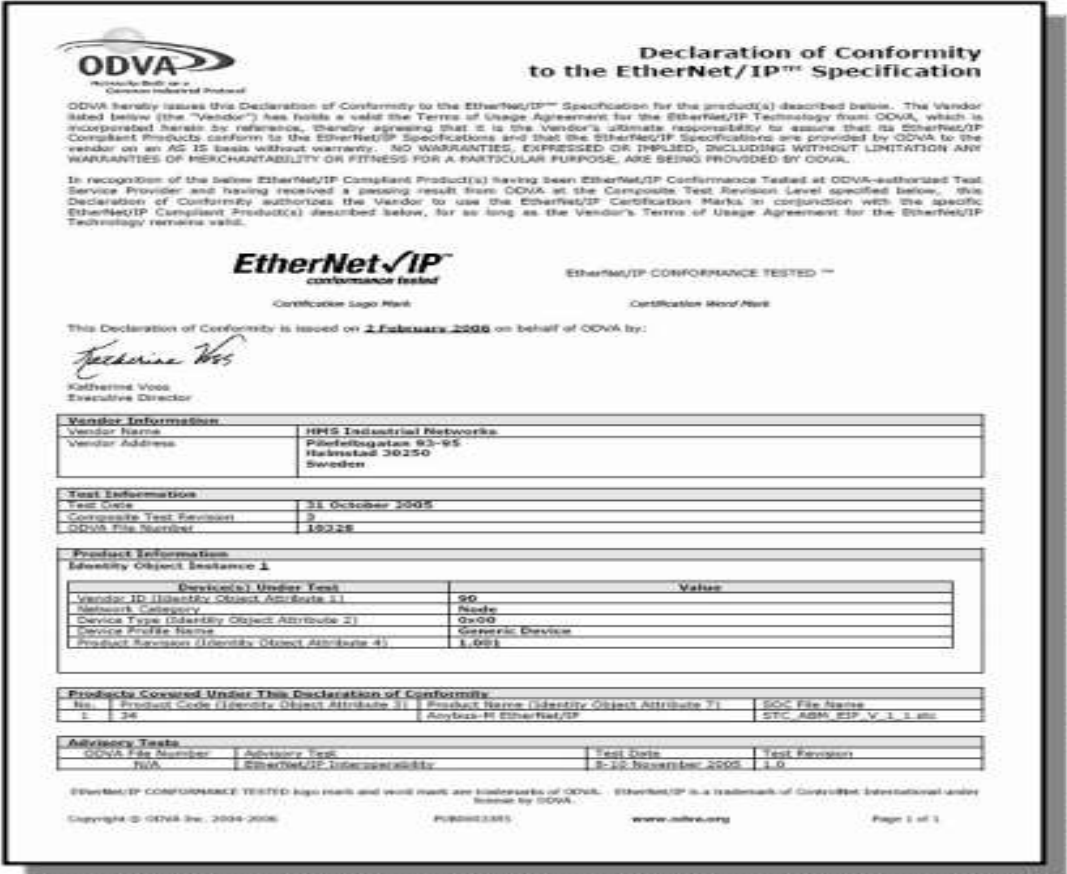
Con un conmutador, los mensajes enviados por Ethernet quedan almacenados provisionalmente en la memoria del propio conmutador. Empleando la Ethernet conmutada, los usuarios pueden efectuar la transición de una tecnología de mensajería estadística a una de conmutación basada en hardware, que tiene una respuesta previsible en el peor de los casos.

4 DESARROLLO DEL PROTOCOLO CIP

Al terminar con la descripción del protocolo CIP para EtherNet/IP y reseñar los sistemas embebidos, el siguiente paso es escoger los elementos que se van a desarrollar en el proyecto, el cual consiste en implantar este protocolo en PC y en un sistema embebido y realizar una comunicación punto a punto entre estos dos dispositivos.

Para que un producto EtherNet/IP pueda operar industrialmente, este deberá tener una certificación denominada “Declaración de conformidad de la especificación EtherNet/IP”, que la otorga la ODVA/CI para que los proveedores obtengan el derecho a utilizar en sus productos la marca de certificación “Prueba de conformidad con EtherNet/IP superada”, que es como finalmente estos fabricantes son avalados y reconocidos

FIGURA 16: Ejemplo de una “Declaración de conformidad de la especificación EtherNet/IP”



Fuente: Tomado de [16], 17 de Noviembre de 2012

En la página [16] se pueden encontrar el listado de las empresas que tiene esta especificación con su respectiva EDS. Actualmente hay más de 240 fabricantes de productos EtherNet/IP.

El proyecto se hizo de acuerdo a la información proporcionada por los dos volúmenes CIP, ya que el propósito para este proyecto es hacer un módulo de aprendizaje del protocolo CIP y no algo certificado industrialmente.

La tarjeta que se utilizó fue una Netduino Plus (ND+), la cual es una plataforma abierta basada en Microsoft.NET Micro Framework, programable en Microsoft Visual C#.

En el Anexo A y B se mostrará cómo opera esta plataforma.

4.1 Elementos escogidos del protocolo CIP, para la realización del proyecto:

El protocolo CIP está basado en modelo objetos, que definen las características propias de un dispositivo.

Todo nodo CIP está compuesto de:

- Clase de objeto.
- Objetos instancia.
- Atributos de instancia.
- Valores de atributo

Todo dispositivo que opere en EtherNet/IP, mínimo implementara los siguientes objetos:

- Objeto Identity.
- Objeto TCP/IP Interface .
- Objeto Ethernet Link .
- Objeto Message Router.
- Objeto Connection Manager.

Se mencionó que generalmente para abrir una conexión CIP en EtherNet/IP, se hace mediante una función UCMM.

Esta se lleva a cabo dentro del servicio Forward_Open (opcional) que está en el objeto Connection Manager.

Pero la forma como opera una conexión entre el PC y la tarjeta (ver anexos), es de forma activa y conectada, por lo que no se puede implementar esta función en el proyecto.

Así que los otros objetos si fueron implementados, en los dos dispositivos.

Además en la tarjeta se implementó un objeto adicional más (para actividad académica) llamado Netduino, que tiene como objetivo hacer parpadear el led de la tarjeta.

Los atributos de claseno fueron implementados porque estos son de tipo opcional.

El objeto Netduino tendrá un código de clase 100, que está en el rango de direcciones permitido para una DEF.

El Path del objeto Message Router, se encargará de enrutar los mensajes explícitos entre diferentes objetos de los dispositivos. Además este objeto presenta atributos de instancia opcionales, por lo que no están implementados.

El paquete de encapsulación de una trama CIP para EtherNet/IP, consta de una serie de comandos. Así que los comandos que se utilizaron para implantar el CIP fueron:

- RegisterSession (para abrir la conexión). Que se aplicó, según la forma de operación de los sockets del C#.
- SendUniData (para transmitir mensajes con conexión).
- UnregisterSession (finalizar la conexión).

4.1.1 Valores asignados de los atributos de instancias del PC y la ND+:

4.1.1.1 Atributos de instancia de la ND+:

Atributos Objeto Identity:

- Vendor ID: 2000.
- Device Type: 120
- Product Code: 2020.
- Major Revision:2
- Minor Revision:1.
- Status: 0.
- Serial Number: ND1.
- Product Name: Netduino Plus.

Atributos Objeto TCP/IP Interface:

- Status:1.
- Configuration Capability:2.
- Configuration Control:2.
- Physical Link Object: 20f62401.
- IPAddress: 192.168.1.100
- Name server: 172.16.255.200
- Name Server 2: 0.0.0.0
- Host Name: Netduino.

Atributo EtherNet Link:

- Interface Speed:1.433 Mbbps
- Interface Flags: 0.

- Physical Address:5C-86-4A-00-2A-81.

4.1.1.2 Atributos de instancia de la PC:

Atributos Objeto Identity:

- Vendor ID: 2050.
- Device Type: 110
- Product Code: 2070.
- Major Revision:2
- Minor Revision:1.
- Status: 0.
- Serial Number: F45U
- Product Name: Axus.

Atributos Objeto TCP/IP Interface:

- Status:1.
- Configuration Capability:2.
- Configuration Control:2.
- Physical Link Object: 21f62401.
- IPAddress: 192.168.1.100
- Name server: 192.168.1.1
- Name Server 2: 0.0.0.0
- Host Name:Equipo-PC.

Atributo EtherNet Link:

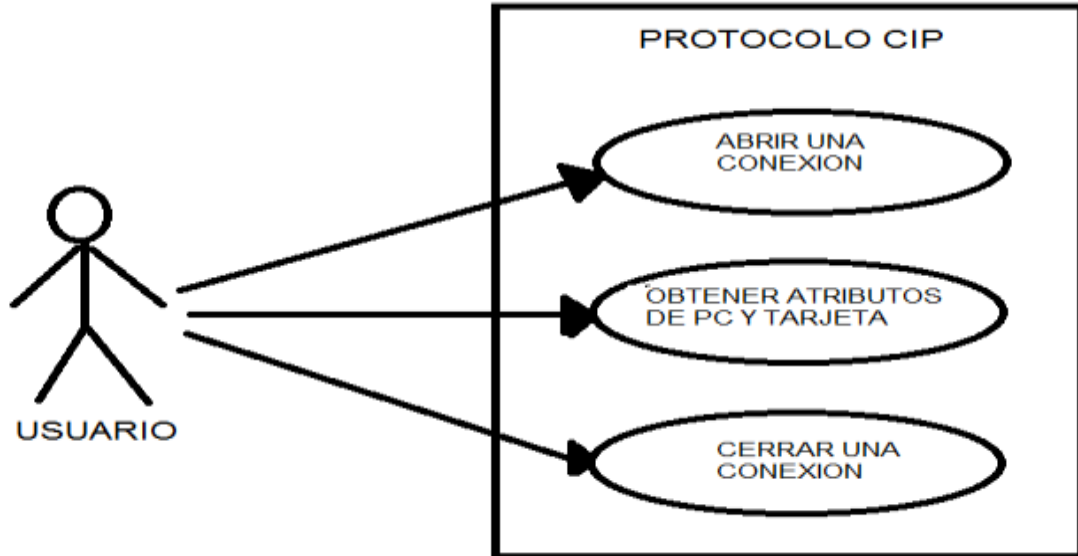
- Interface Speed:1.433 Mbbps
- Interface Flags: 0
- Physical Address:DC-85-DE-14-6B-B4

Los atributos de ambos dispositivos se acceden cada uno mediante el path del Objeto Message Router, con lo cual se implementó el servicio Get_Attribute_Single.

Veamos los diagramas UML, para comprender el funcionamiento del programa.

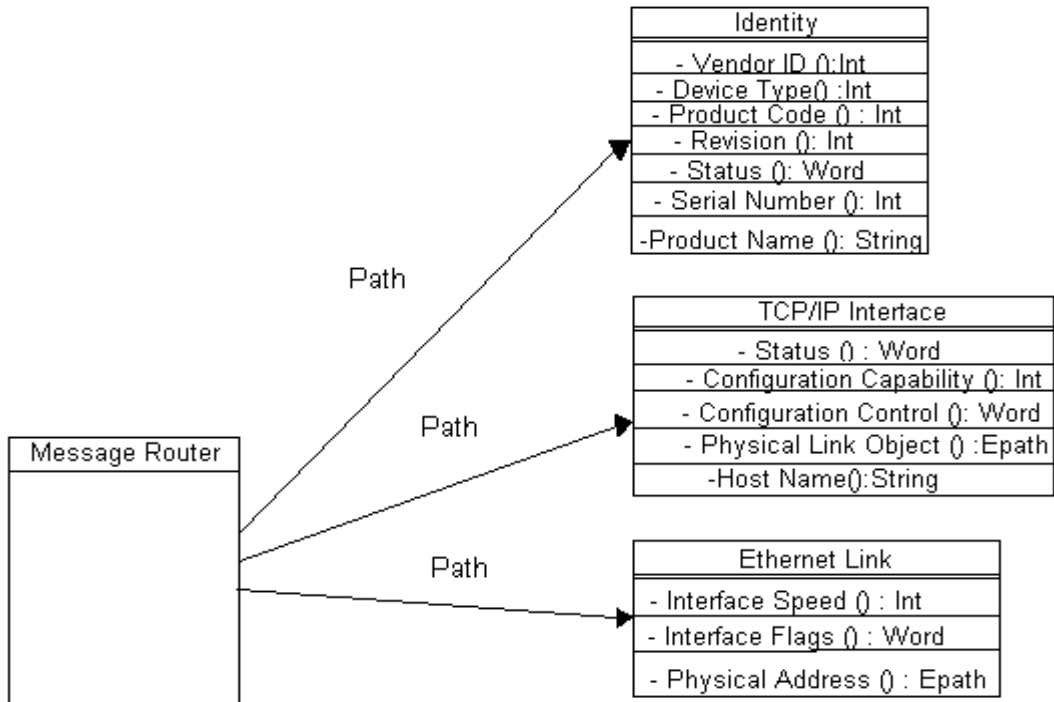
4.2 Diagramas UML del software del protocolo CIP:

Figura 17: Diagrama de caso de uso



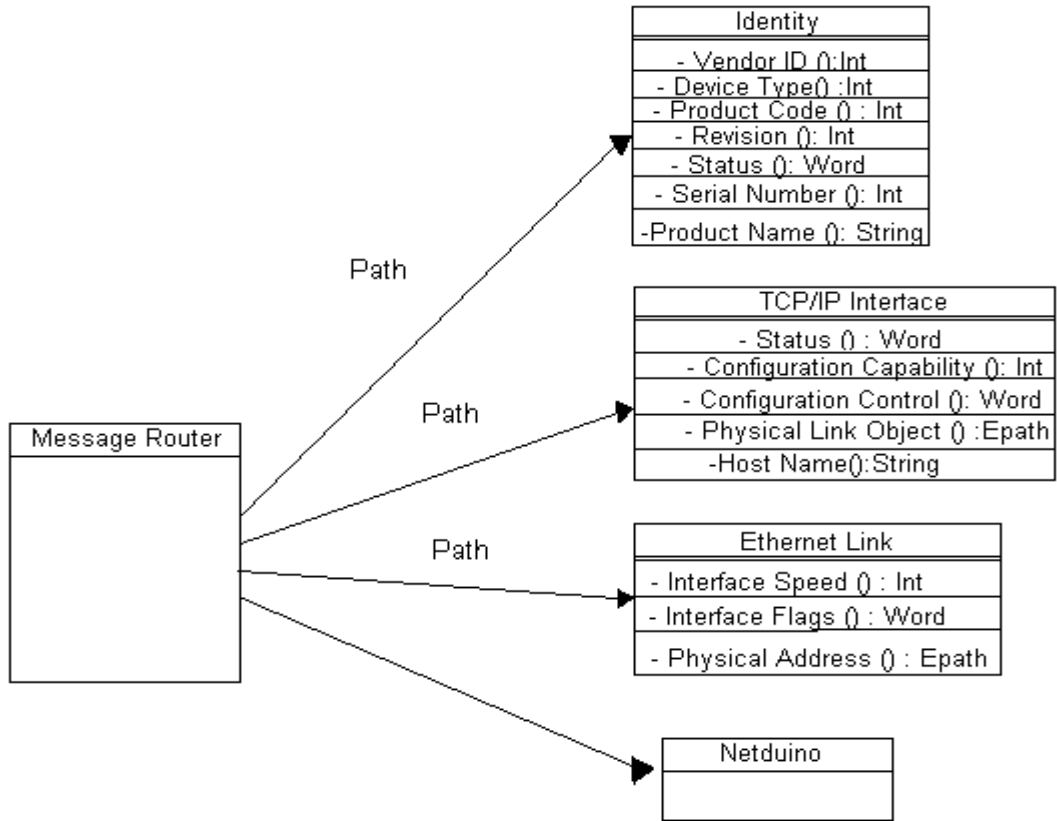
Fuente propia

Figura 18: Diagrama de objetos para los atributos del PC



Fuente propia

Figura 19: Diagrama de objetos para los atributos de la Netduino Plus.



Fuente propia

4.3 Formato SendUnidata utilizado para este proyecto:

Tabla 54: Comando de encapsulación utilizado para este proyecto

Cabecera de Encapsulación	Nombre del campo	Valor
	Status	0
	Options	0

	Sender Context	Path del Message Router
Comando de datos específicos	Tipo ID	A1

Fuente propia

4.4 Vista del protocolo CIP implantado:

Finalmente, el diseño del protocolo CIP realizado en este proyecto se hizo de dos formas:

- Una solución (proyecto en Windows Form en C#) grafica desde la cual un usuario, ubica el objeto, código de clase, código de atributo y el packet epath (path del message router) con el cual se accede al atributo de instancia deseado ya sea del PC o de la ND+.
- Una solución de escritorio (proyecto en modo consola en C#), desde la cual el usuario colocando el valor del path, activa y obtiene el valor de atributo de instancia deseado del PC o ND+.

Por ejemplo aquí vamos a acceder al atributo Vendor ID, del objeto Identity de la ND+.

Figura 20: Direccionamiento CIP para el atributo Vendor ID de la ND+

Fuente propia

Figura 21: Ubicación del atributo Vendor ID de la ND+

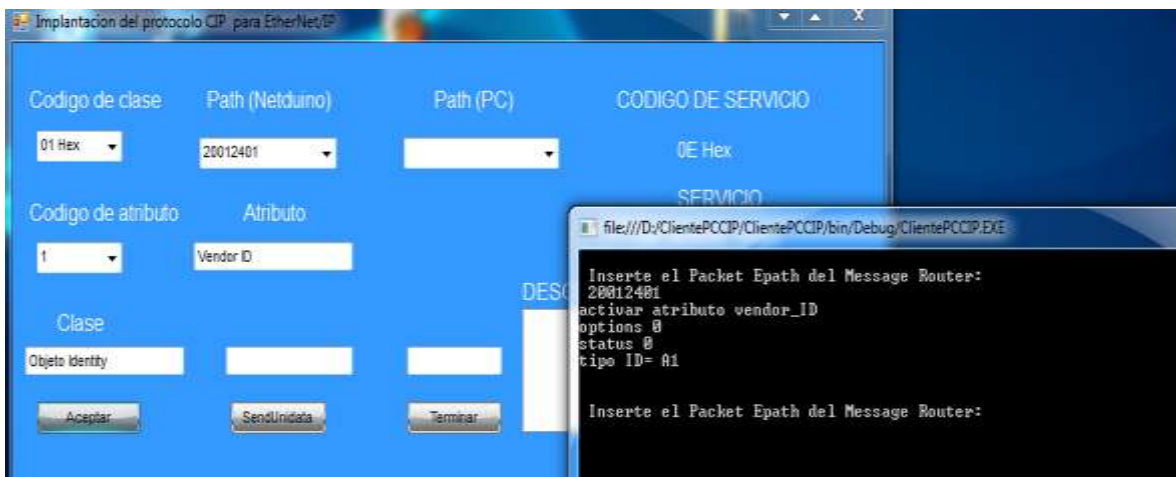


Fuente propia

Al darle aceptar, se activara el el nombre de la clase y del atributo y valor del path en el campo de relleno del SendUniData, y oprimiendo dicho botón, se activará el valor deseado de instancia tanto del PC o dela ND+. En el proyecto en modo consola solo se colocara el valor del Path, para el acceso a un atributo de instancia del PC o de la ND+.

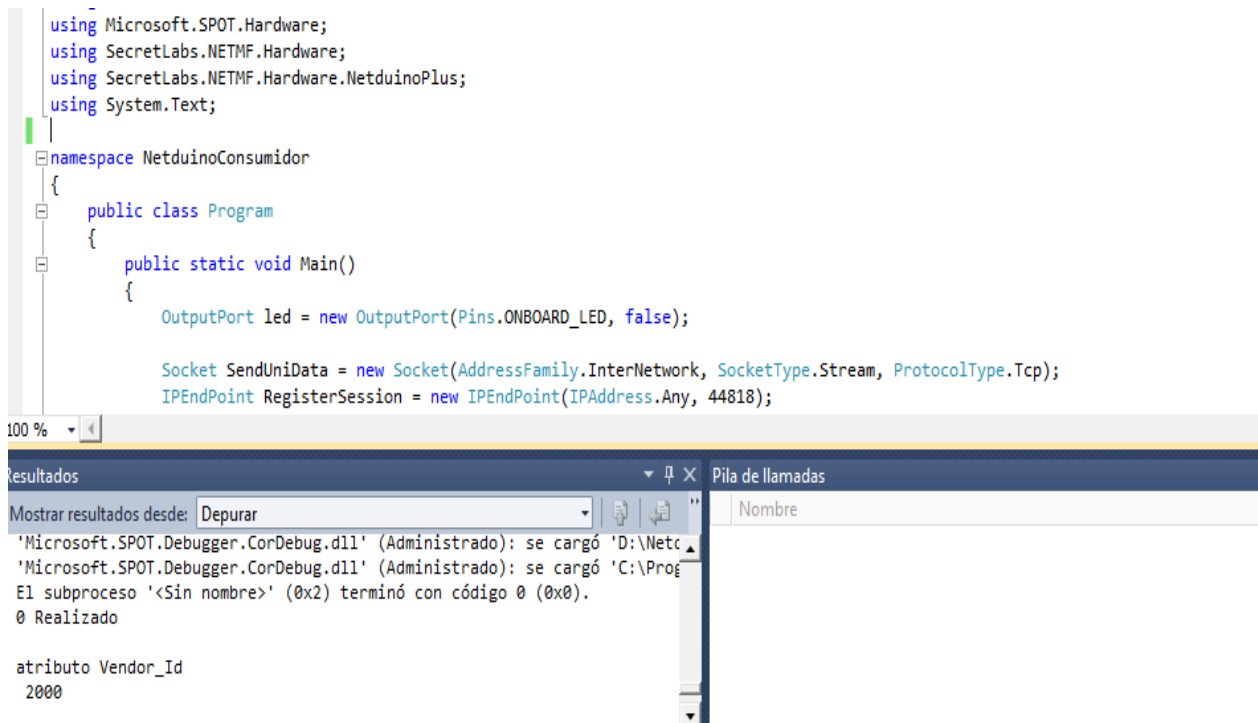
Así que ejecutando el proyecto en modo consola, y colocando el path respectivo para este atributo, se obtiene las siguientes respuestas.

Figura 22: Ubicación del atributo Vendor ID en modo consola para la ND+



Fuente propia

Figura 23: Ubicación del atributo Vendor ID en modo consola para la ND+



Fuente propia

Aquí podemos ver claramente en los dos gráficos los formatos que se exigen de solicitud y respuesta tanto del Message Router y del comando SendUniData. Ahora miremos el funcionamiento para los atributos de instancia del PC.

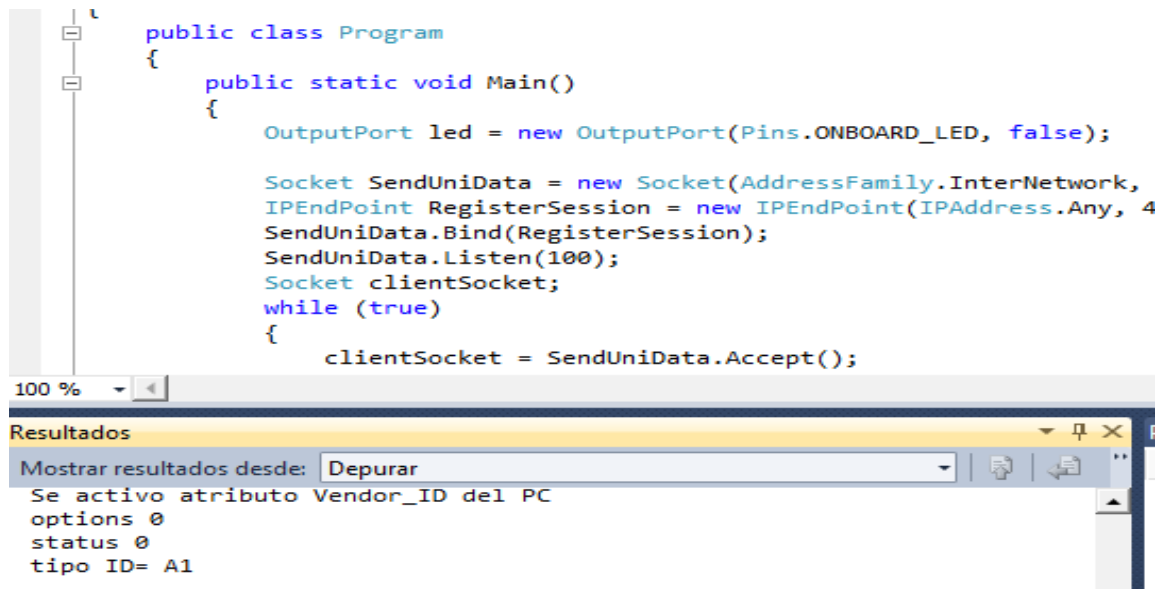
Figura 24: Direccionamiento CIP para el atributo Vendor ID para el PC



Fuente propia

Al colocar el path en el modo consola obtenemos en el proyecto de la ND+

Figura 25: Activación del objeto identity del PC



```
public class Program
{
    public static void Main()
    {
        OutputPort led = new OutputPort(Pins.ONBOARD_LED, false);

        Socket SendUniData = new Socket(AddressFamily.InterNetwork,
        IPEndPoint RegisterSession = new IPEndPoint(IPAddress.Any, 4
        SendUniData.Bind(RegisterSession);
        SendUniData.Listen(100);
        Socket clientSocket;
        while (true)
        {
            clientSocket = SendUniData.Accept();
```

Resultados

Mostrar resultados desde: Depurar

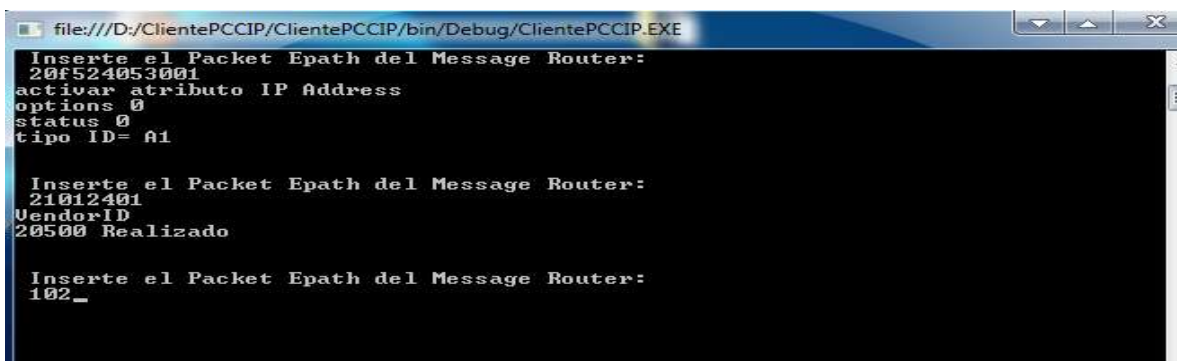
Se activo atributo Vendor_ID del PC
options 0
status 0
tipo ID= A1

Fuente propia

Y así sucesivamente podemos obtener los valores de atributos de instancia de cada dispositivo.

Finalmente para terminar la sesión de ambos dispositivos, lo hacemos mediante el comando UnRegisterSession que tiene código hexadecimal 66, en este caso el 102 decimal.

Figura 26: Finalización de la conexión



```
file:///D:/ClientePCCIP/ClientePCCIP/bin/Debug/ClientePCCIP.EXE
Inserte el Packet Epath del Message Router:
20f524053001
activar atributo IP Address
options 0
status 0
tipo ID= A1

Inserte el Packet Epath del Message Router:
21012401
VendorID
20500 Realizado

Inserte el Packet Epath del Message Router:
102_
```

Fuente propia

De esta manera se logra una comunicación CIP punto a punto entre el PC y el sistema embebido, teniendo en cuenta los requerimientos de este protocolo, visualizando los atributos de instancia de ambos dispositivos, y transmitiéndolos por el puerto AF12, que es el único puerto que garantiza una comunicación EtherNet/IP.

4.5 EDS de la PC y la Netduino Plus:

Figura 27: EDS Netduino Plus

```
[File]
  DescText="Netduino EDS File";
  CreateData=21-12-2012;
  CreateTime=9:44:38;
  Revision= 2.1;
[Device]
  VendCode= 2000 ;
  VendName= 120 ;
  ProdType= 0;
  ProdTypeStr= "Generic" ;
  MajRev= 2;
  MinRev= 1;
  ProdName= "Netduino plus" ;
[DeviceClassification]
  Class1=Ethernet/IP;
[Port]
  TCP,
  "Ethernet/IP port";
  "20 F5 24 01";
```

Fuente propia

Figura 28: EDS PC

```
[File]
  DescText="Asus EDS File";
  CreateData=21-12-2012;
  CreateTime=12:27:14;
  Revision= 2.1;
[Device]
  VendCode= 2050 ;
  VendName= 110 ;
  ProdType= 0;
  ProdTypestr= "Generic" ;
  MajRev= 2;
  MinRev= 1;
  ProdName= "Asus" ;
[DeviceClassification]
  Class1=Ethernet/IP;
[Port]
  TCP,
  "Ethernet/IP port";
  "21|F5 24 01";
```

Fuente propia

CONCLUSIONES

- Se logró describir de manera básica y precisa el protocolo CIP adaptado a EtherNet/IP en español, gracias a un proceso de traducción. Esta información se encuentra detalladamente descrita en inglés, lo cual puede beneficiar a partir de éste momento a cualquier estudiante del programa de Ingeniería en Automática Industrial (PIAI), que desee explorar el funcionamiento fundamental de esta especificación.
- Se logró una comunicación punto a punto entre un PC y un sistema embebido, implantando el protocolo CIP para EtherNet/IP, en ambos dispositivos. Este alcance permite que dicha comunicación se pueda utilizar como recurso de aprendizaje en la enseñanza de redes industriales para el PIAI.
- El presente proyecto plantea un punto de partida para el desarrollo de proyectos de comunicación en sistemas embebidos y también de ejecución de otras aplicaciones que se pueden realizar tanto en la plataforma Netduino Plus como en las otras tarjetas que ofrece la comunidad Netduino.
- La tarjeta que se utilizó para este proyecto, no puede transmitir datos, a otro dispositivo porque se mostró inestable en cuanto a su configuración de parámetros IP (estática y dinámicamente).
- Aunque el protocolo CIP es abierto y estándar. Las empresas fabricantes de EtherNet/IP poseen un sistema de encriptación de datos riguroso, lo cual hace que su sistema de transmisión de datos no sean vulnerados y alterados.

TRABAJOS FUTUROS

- Diseño e implementación de una arquitectura EtherNet/IP para el Laboratorio de Control de Procesos del PIAI.
- Realizar un control de movimiento multieje a través de un sistema empotrado, implantando EtherNet/IP con tecnología CIP MOTION.
- Implantar el protocolo CIP sobre una plataforma embebida que soporte CAN, para el montaje de una red DeviceNet. Y hacer bridging y un ruteo de mensajes CIP con lo realizado en este proyecto.
- Escenarios de automatización donde todos los dispositivos de las plantas del laboratorio del control de proceso (LCP) del PIAI se encuentren conectados a una red (DeviceNet, EtherNet/IP, ControlNet o CompoNet).
- Planteamiento de una arquitectura NetLinx para el LCP.
- Planteamiento de una arquitectura de red DeviceNet Safety para el LCP.

BIBLIOGRAFIA

- [1] A.JAVIER BARRAGAN PIÑA. “7 Aplicaciones en la industria”,2012. Disponible en <http://uhu.es/antonio.barragan/content/7-aplicaciones-industria> [Acceso 10 de Septiembre de 2012].
- [2] WELLINGTON ANTONIO REMACHE BENAVIDES, “Diseño de una red EtherNet/IP para la implantación en ambiente industrial”, 2009. Tesis de pregrado, Facultad de Ingeniería Electrónica, Escuela Politécnica del Ejercito (Ecuador). Disponible en <http://www3.espe.edu.ec:8700/handle/21000/482>[Acceso 10 de Septiembre de 2012].
- [3] SMAR, “Tutorial DeviceNet”2011. Disponible en <http://www.smar.com/espanol/devicenet.asp>.
- [4] Luis Suazo “Red ControlNet” 2010 [Online]. Disponible en <http://automatizacion.bligoo.com/content/view/900361/Red-ControlNet.html>
- [5] Oscar Mena “CompoNet la nueva red de sensores y actuadores”.Disponible en www.industriaaldia.com/articulos/65-16.pdf .
- [6] EDITORIAL CYPSELA S.”EtherNet/IP, la red industrial abierta y estándar”. Disponible en http://www.cypsela.es/revista/revistas_anteriores/revista_217/articulos/art3.html [Acceso 11 de Septiembre de 2012].
- [7] SIEMON “Ethernet/IP” Disponible en http://www.siemon.com/la/white_papers/03-10-13-ethernet-ip.asp[Acceso 12 de Septiembre de 2012].
- [8]Juan Maximiliano Muñoz“Estudio de aplicación de los estándares DeviceNet y ControlNet de comunicaciones industriales como solución de red de campo y proceso en una planta industrial” 2007. Tesis de pregrado. Facultad de Ciencias de la Ingeniería. Universidad Austral de Chile. Disponible en <http://cybertesis.uach.cl/tesis/uach/2007/bmfcim971e/doc/bmfcim971e.pdf>
- [9]ControlNet Internacional and Open Device Vendor Association “The CIP Networks Library volume1”2007.Edition 3.3.Noviembre de 2007ftp://ftp.heapg.com/AB-OCS/Ethernet%20IP%20CD/CIP/Vol1_3.3.pdf [Acceso de Septiembre de 2012 a Diciembre de 2012].
- [10] Odva “The Common Industrial Protocol (CIP™) and the Family of CIP Networks” 2006.

Disponible en http://www.odva.org/Portals/0/Library/Publications_Numbered/PUB00123R0_Common%20Industrial_Protocol_and_Family_of_CIP_Netw.pdf [Acceso 15 de Septiembre de 2012].

[11] Lenze AC Tech Corporation “Módulo de comunicaciones SMVector EtherNet/IP”. 2009. [Acceso 28 de Septiembre de 2012].

[12] ControlNet Internacional and Open Device Vendor Association “The CIP Networks Library volume 2, Ethernet/IP adaptation of CIP” Edición 1.4. Noviembre 2007 Disponible ftp://ftp.heapg.com/AB-OCS/Ethernet%20IP%20CD/EtherNet-IP/Vol2_1.4.pdf [Acceso de Septiembre de 2012 a Diciembre de 2012].

[13] “ICO (Windows icon)”. Disponible en <http://www.flavionet.com/lex/ico.htm> [Acceso 15 de Noviembre de 2012].

[14] ¿Qué es un sistema embebido? Disponible en <http://www.idose.es/faq/49-sistemas-embebidos/54-que-es-un-sistema-embebido>. [Acceso 15 de Noviembre de 2012].

[15] Antonio Nadal Galiana “Sistemas embebidos”. Disponible en <http://server-die.alc.upv.es/asignaturas/PAEEES/2005-06/A07%20-%20Sistemas%20Embebidos.pdf>. [Acceso 16 de Noviembre de 2012].

[16] ODVA. “EtherNet/IP DOCs” Disponible en <http://www.odva.org/Home/CIPPRODUCTCOMPLIANCE/DeclarationsofConformity/EtherNetIPDOCs/tabid/159/Ing/en-US/Default.aspx>. [Acceso 17 de Noviembre de 2012].

[17] Virtual Market Disponible en http://www.virtualmarket.com.ve/product.php?id_product=17.2012.

[18] Plug and Embedded Computers. Disponible en <http://plugcomp.com/>

[19] Dynamo Electronics. Disponible en http://www.dynamoelectronics.com/dynamo-tienda-virtual.html?page=shop.product_details&category_id=82&flypage=dynamo.tpl&product_id=252

[20] GE Tech Wiki “Arduino Ethernet shield”. Disponible en http://www.geeetech.com/wiki/index.php/Arduino_Ethernet_Shield

[21]Solo Electrónicos “Plataforma Netduino Plus”. Disponible en <http://solo-electronicos.blogspot.com/2012/01/plataforma-netduino-plus.html>

[22]10 Rem.Net. Disponible en <http://10rem.net/blog/2010/10/03/netduino-netduino-plus-fez-panda-arduino-deumilanove>

[23] Francisco Javier Ceballos “Enciclopedia de Microsoft, Visual C#”. 2006. Enciclopedia Alfaomega.

[24]Visual Studio Express. Disponible en <http://www.microsoft.com/visualstudio/eng/products/visual-studio-express-products>

[25]Microsoft Centro de descargas. Disponible en <http://www.microsoft.com/en-us/download/details.aspx?id=8515>

[26]Netduino. Disponible en <http://netduino.com/netduinoplus/specs.htm>

[27]MY NETDUINO “Netduino Webserver 1.0” Disponible en <http://mynetduino.blogspot.com/>