SERVICIOS TELEMÁTICOS AVANZADOS

PROTOCOLOS DOMÓTICOS

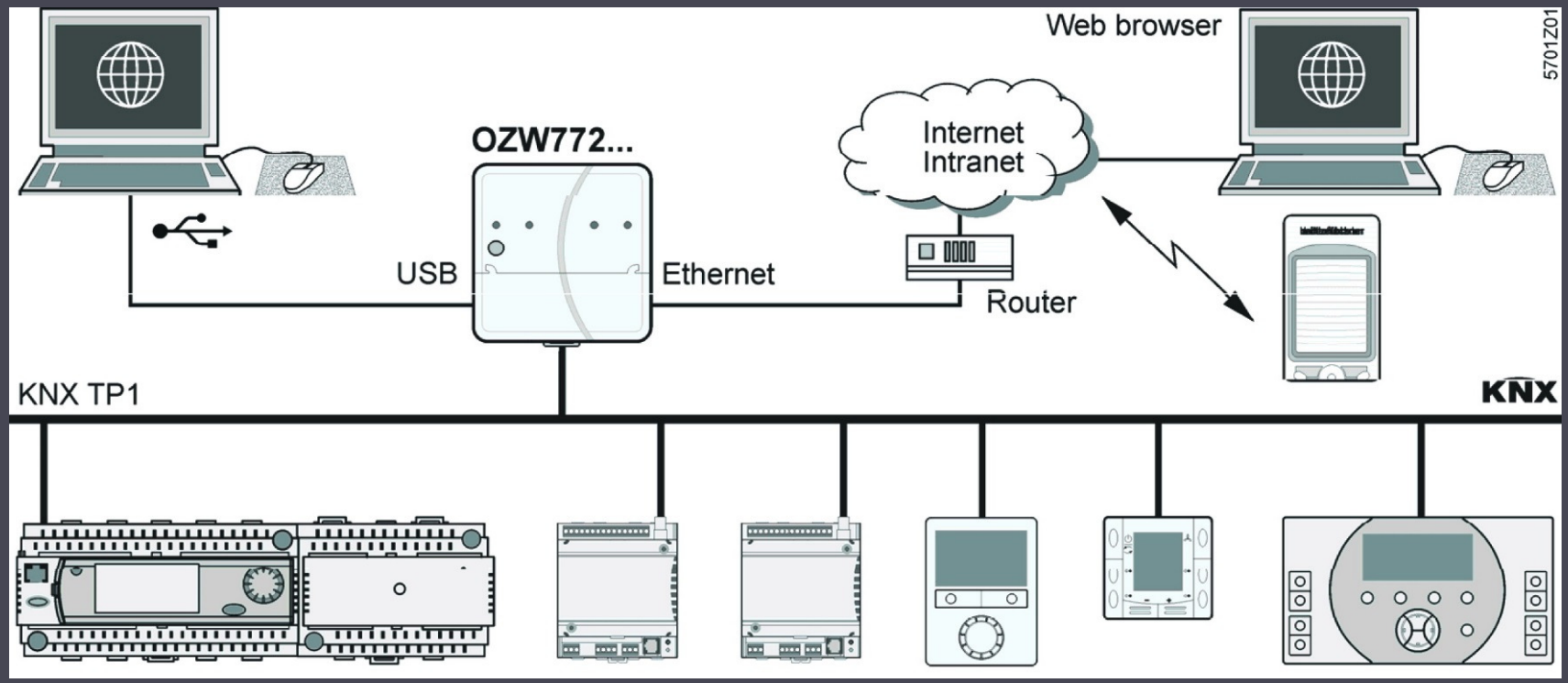
Ander Gabilondo Areta

SERVICIOS TELEMÁTICOS AVANZADOS

SERVICIOS TELEMÁTICOS AVANZADOS

1.- INTRODUCCIÓN

¿Por qué existen los protocolos domóticos?



Porque es imprescindible una comunicación cuando el entorno es de tipo distribuido - > EL PODER DE LAS TELECOMUNICACIONES

SERVICIOS TELEMÁTICOS AVANZADOS

1.- INTRODUCCIÓN

Conoceremos protocolos de comunicación en sistemas domóticos

¿Para qué?

1. Desarrollo de aplicaciones de supervisión para PCs (SCADAS)
2. Desarrollo de dispositivos domóticos
3. Desarrollo de aplicaciones de control para móviles
4. Generación de aplicaciones de alto nivel añadido

SERVICIOS TELEMÁTICOS AVANZADOS

1.- INTRODUCCIÓN (SCADAS)

Supervisory Control And Data Acquisition

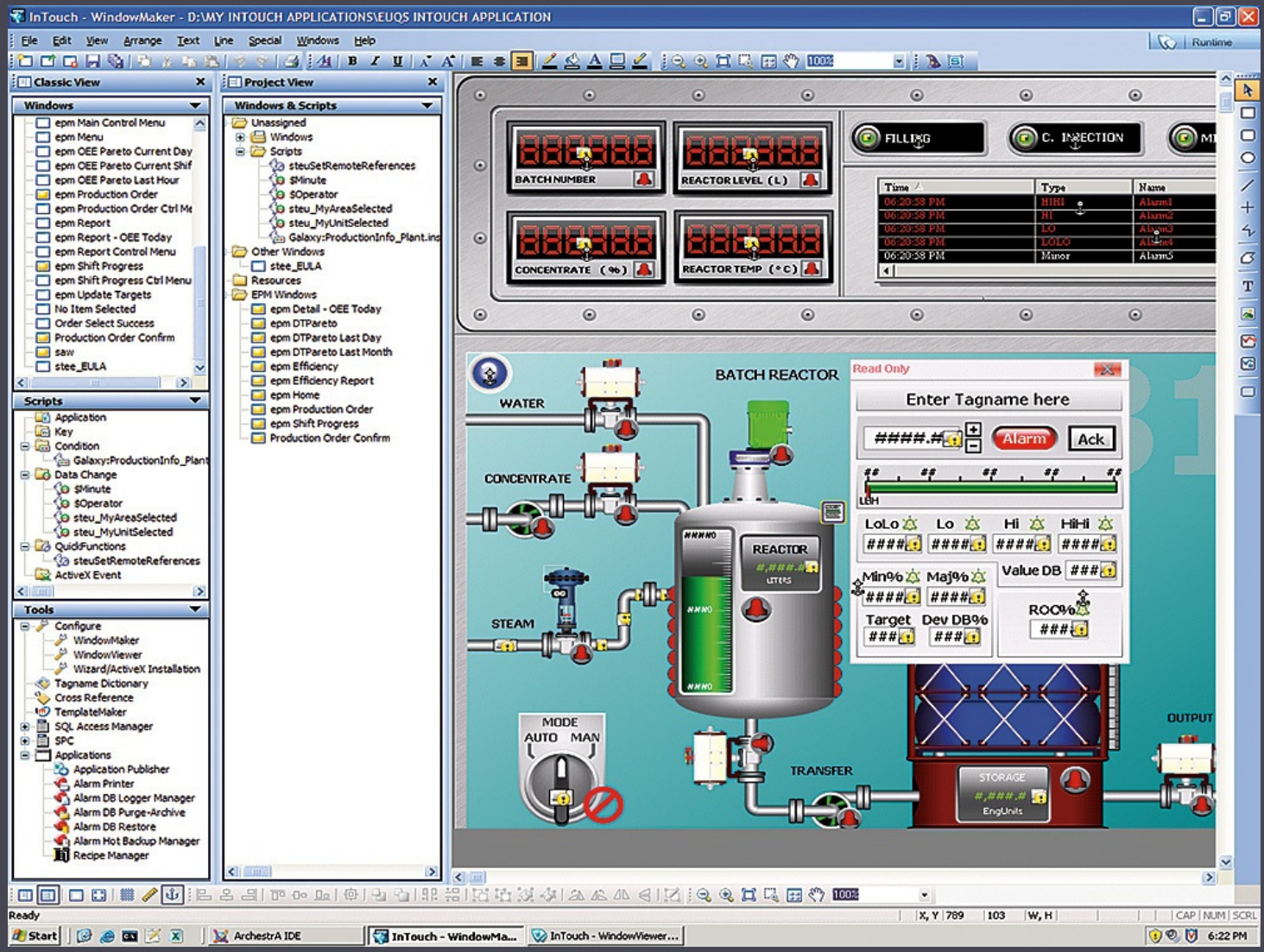
Aplicaciones para PCs que permiten:

1. Envío de órdenes a los dispositivos de una instalación
2. Supervisión de estado de los elementos controlados
3. Almacenamiento de datos históricos (consumo, alarmas etc.)

SERVICIOS TELEMÁTICOS AVANZADOS

1.- INTRODUCCIÓN (SCADAS)

Supervisory Control And Data Acquisition



1.- INTRODUCCIÓN (DISPOSITIVOS)

El desarrollo de dispositivos requiere:

1. Conocimiento del protocolo
2. Conocimientos de electrónica digital
3. Dinero (en caso de certificación)

COSTE

NIVEL DE DESARROLLO

SERVICIOS TELEMÁTICOS AVANZADOS

1.- INTRODUCCIÓN (APLICACIONES PARA MÓVILES)

En los últimos años son las que más han proliferado

1. Interfaces amigables
2. Bajo coste de desarrollo
3. Bajo precio de adquisición
4. Fácil comercialización

SERVICIOS TELEMÁTICOS AVANZADOS

1.- INTRODUCCIÓN (APLICACIONES ALTO NIVEL AÑADIDO)

Los sistemas domóticos y de control tratan la información de forma tradicional

1. Comunicaciones monolenguaje
2. Relaciones causa/efecto y acción/reacción
3. Almacenamiento lineal

¿Qué se puede aportar?

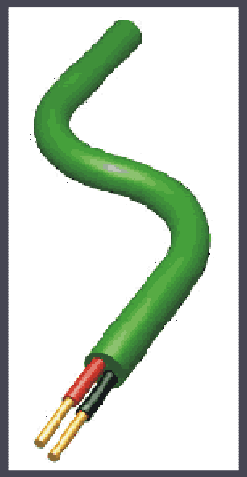
1. Inteligencia artificial
2. Integración
3. Inteligencia ambiental

SERVICIOS TELEMÁTICOS AVANZADOS

2.- MEDIOS DE TRANSMISIÓN

Los medios de transmisión utilizados son los siguientes:

1. Par trenzado
2. Radiofrecuencia
3. Corrientes portadoras
4. Infrarrojo, RS232 etc.
5. **IP (Ethernet)**

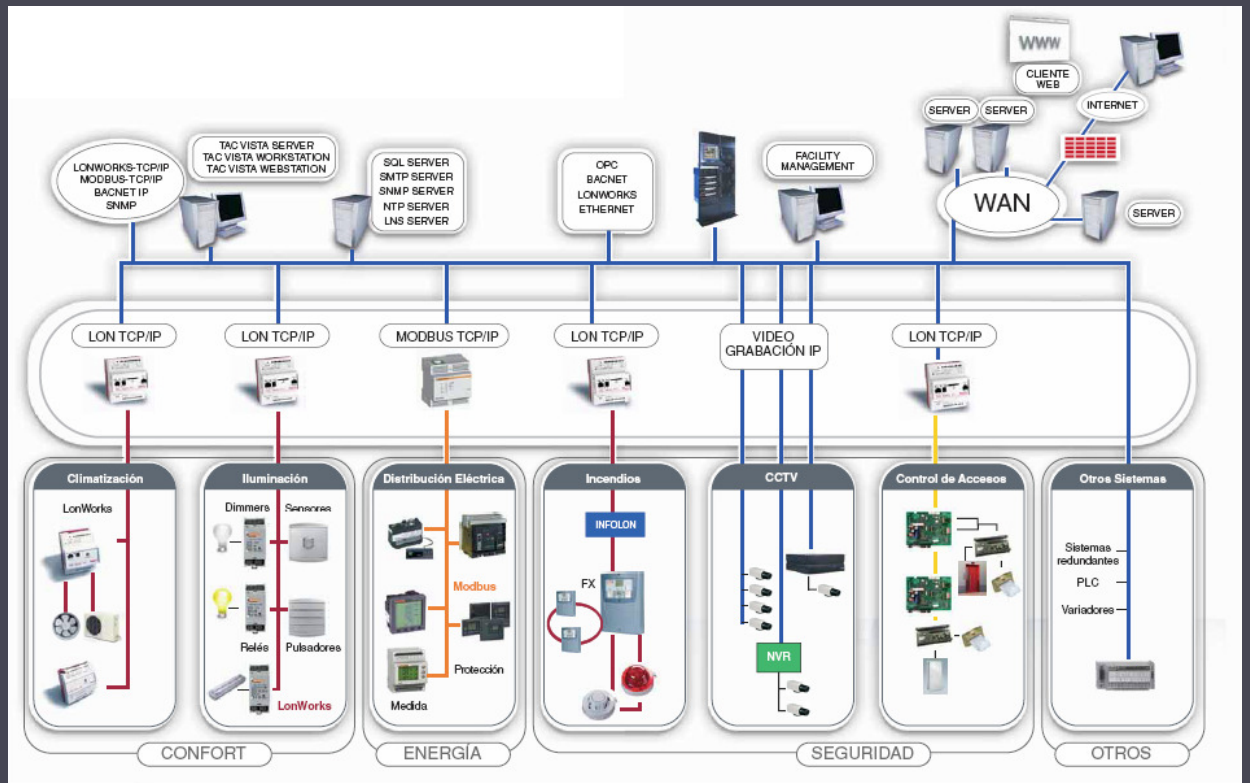


SERVICIOS TELEMÁTICOS AVANZADOS

2.- MEDIOS DE TRANSMISIÓN

Desarrollamos la opción IP porque:

1. Su coste de desarrollo es menor
2. No requiere de desarrollo electrónico
3. Casi todos los protocolos (sistemas) confluyen en esta red INTEGRACIÓN



3.- ESTÁNDARES DE CONTROL

Los principales protocolos estándares de control del momento son:

Modbus®

PARTNER KNX

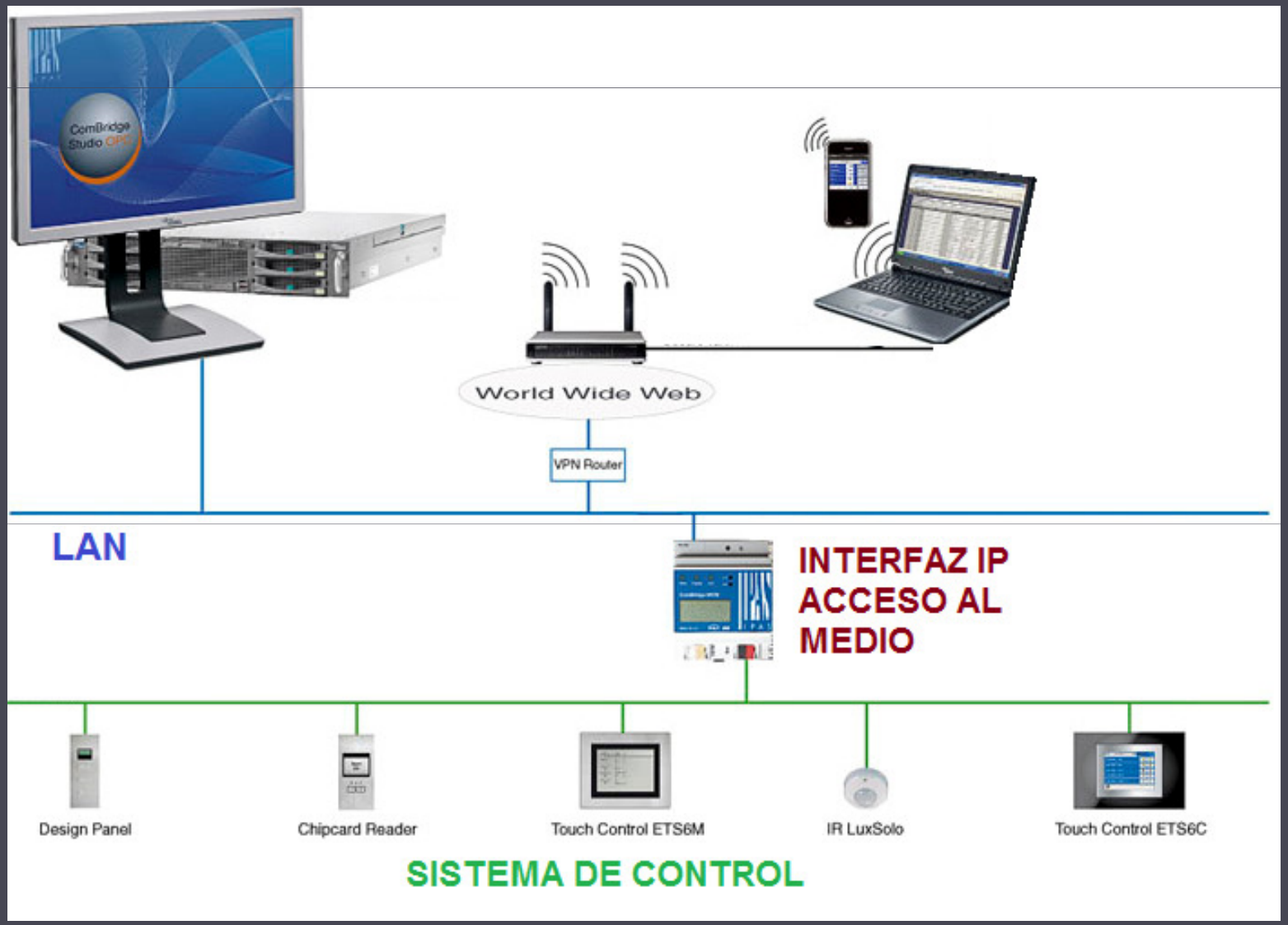
LONWORKS

ASHRAE BACnet™

Todos ellos tienen desarrollada su versión IP

3.- ESTÁNDARES DE CONTROL

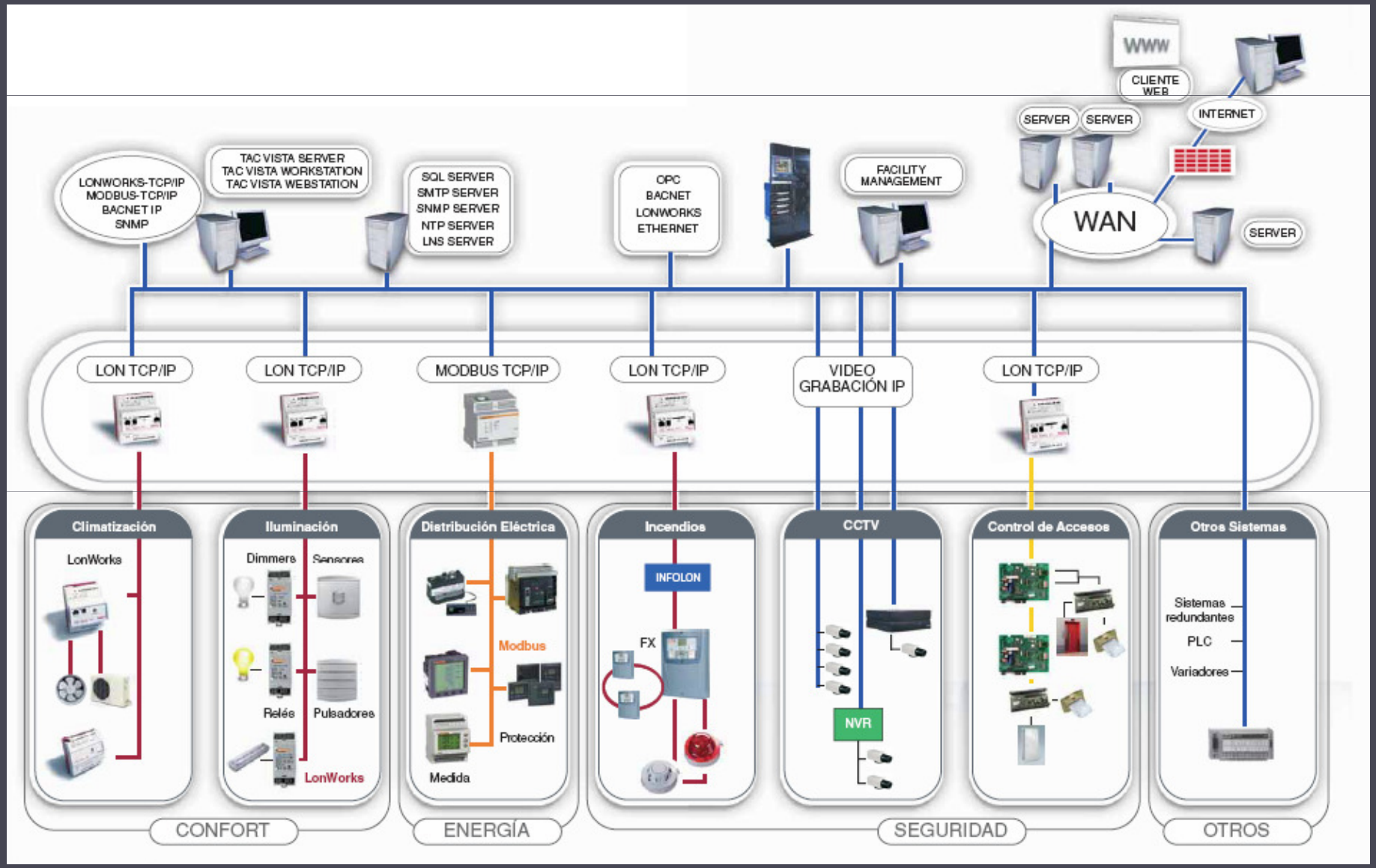
En todos los casos nuestra infraestructura mínima será:



SERVICIOS TELEMÁTICOS AVANZADOS

3.- ESTÁNDARES DE CONTROL

Aunque podría existir más de un sistema:



SERVICIOS TELEMÁTICOS AVANZADOS

3.1- KNXNET/IP

El estándar KNX está documentado en la norma ISO/IEC 14543-3

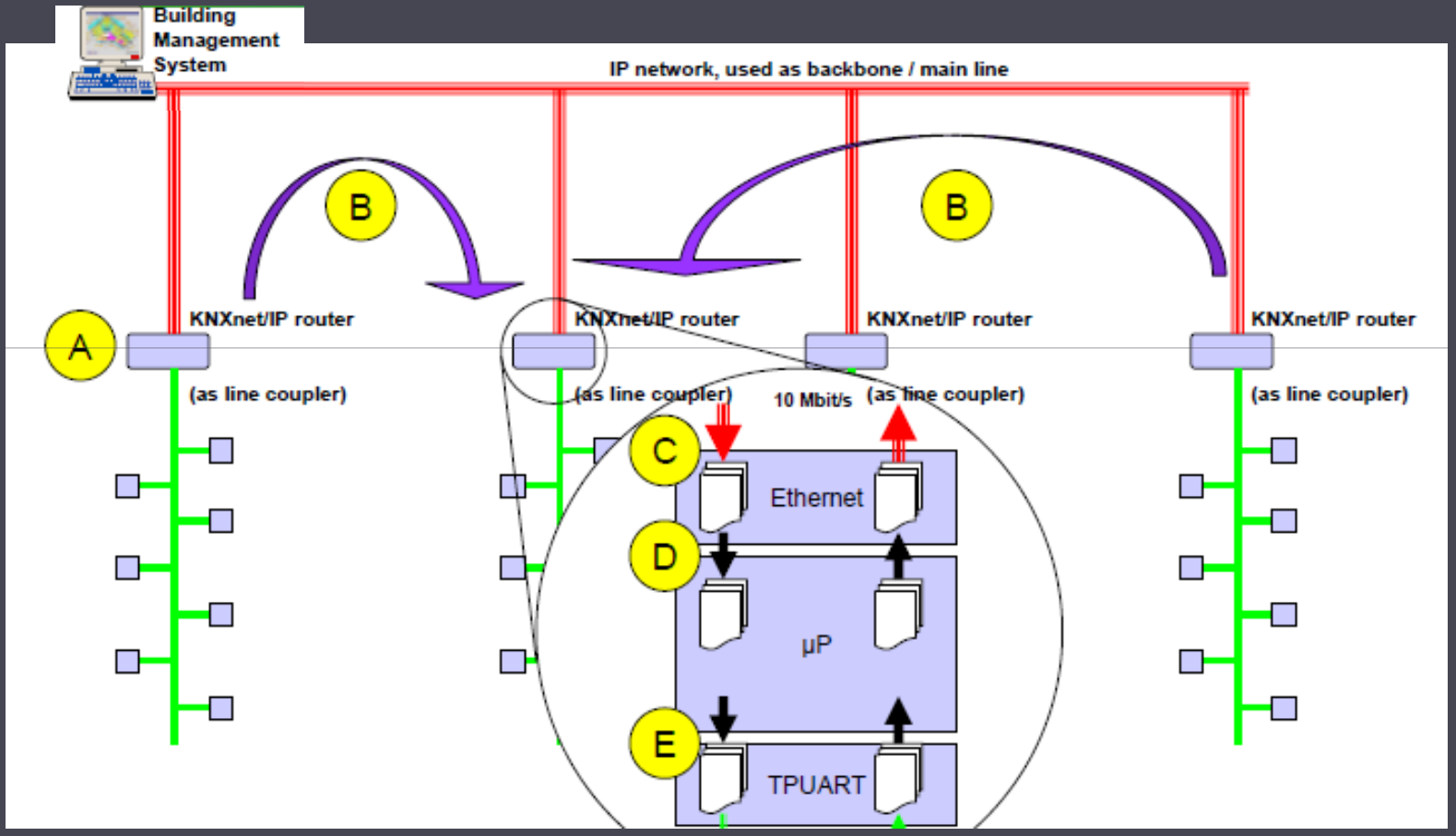
Toda la información está disponible en las “KNX Specifications” (De pago €€€)

http://www.knx.org/uk/knx-standard/knx-specifications/

3.1- KNXNET/IP

Esta parte del estándar está dividida en dos grandes bloques

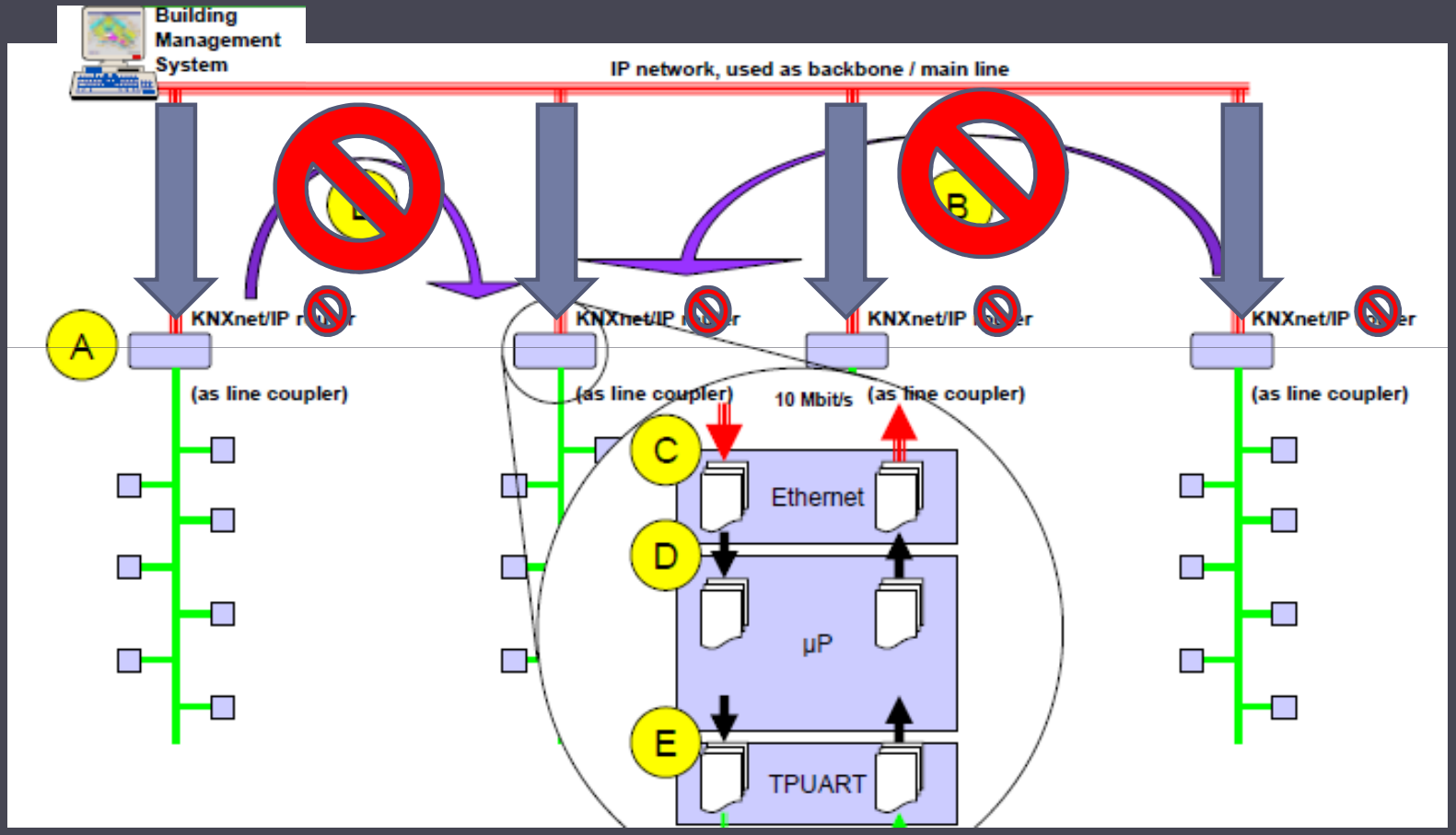
1. **KNXnet/IP Routing (Se requiere un Router)**
2. KNXnet/IP Tunneling (Se requiere un Router o un Interfaz IP)



3.1- KNXNET/IP Tunneling

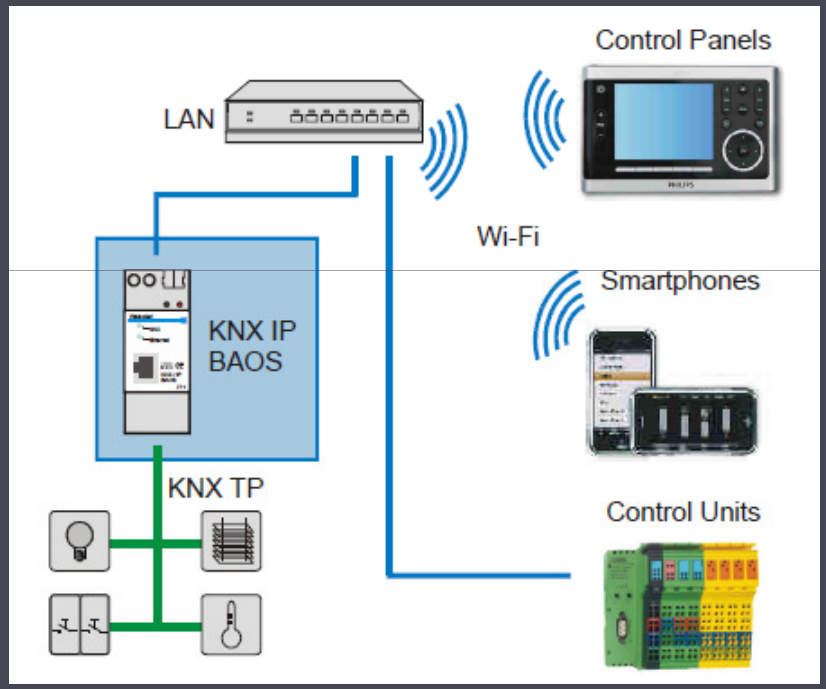
Esta parte del estándar está dividida en dos grandes bloques

1. KNXnet/IP Routing (Se requiere un Router)
2. **KNXnet/IP Tunneling (Se requiere un Router o un Interfaz IP)**



3.1- KNXNET/IP Tunneling

This Chapter 3/8/4 “Tunnelling” of the KNXnet/IP specification describes **point-to-point** exchange of KNX telegrams over an **IP network** between an KNXnet/IP **device** acting as a **server** and an KNXnet/IP **Client** for configuration and diagnostics. KNX frames are encapsulated inside IP datagrams. KNXnet/IP Tunnelling does **not** address **timing issues** caused by IP data network **latency** greater than **one second**.

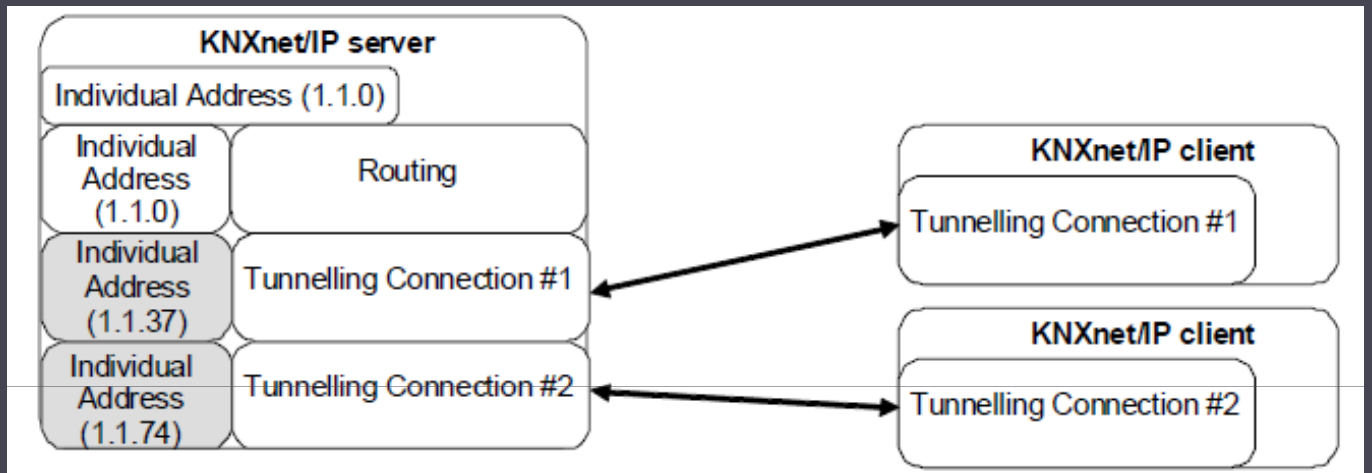


3.1- KNXNET/IP Tunneling

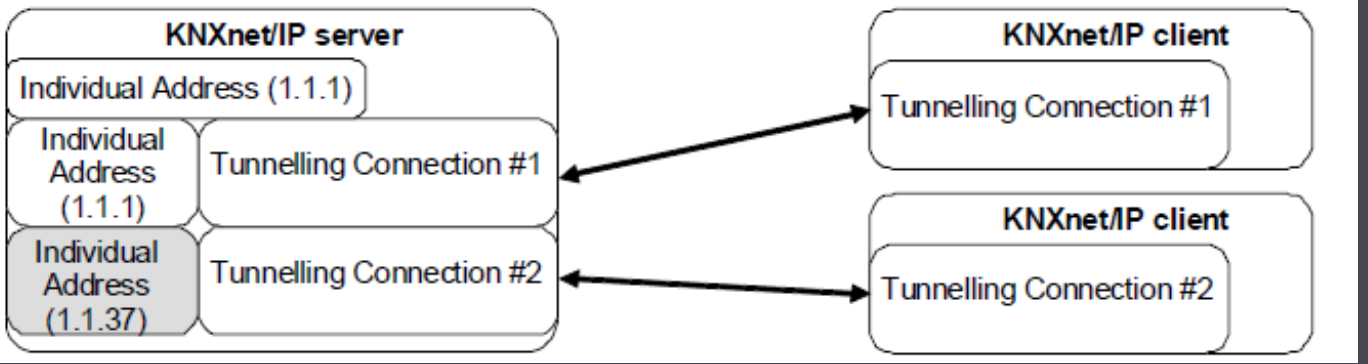
Los interfaces IP de acceso al medio soportan típicamente varias conexiones simultáneas (3-5)

Esto conlleva una separación de las direcciones físicas virtuales

ROUTER



INTERFAZ IP



3.1- KNXNET/IP Tunneling – Header

Los paquetes deben encapsularse en paquetes TCP/IP o UDP cuya información útil es un frame KNXnet/IP

Todos los paquetes transmitidos entre cliente y servidor deben contener un encabezado (Header)

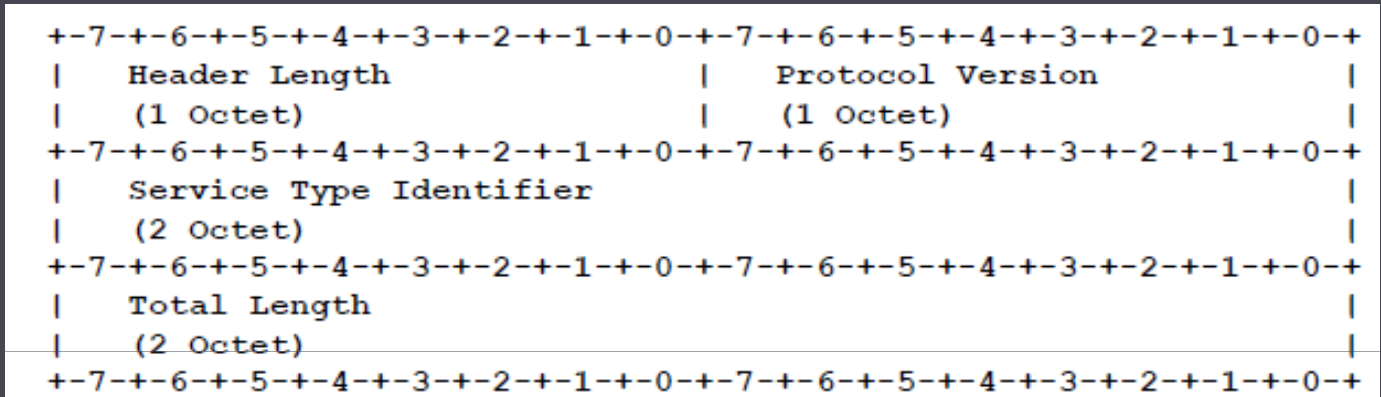


Figure 2 – KNXnet/IP header binary format

Table 3 – Common KNXnet/IP constants

Constant name	Value	V.4)	Description
KNXNETIP_VERSION_10	10h	1	Identifier for KNXnet/IP protocol version 1.0
HEADER_SIZE_10	06h	1	Constant size of KNXnet/IP header as defined in protocol version 1.0

3.1- KNXNET/IP Tunneling – Códigos

Los servicios soportados son muy variados

1. **Servicios Básicos (Core)**
2. Gestión de dispositivos
3. **Tunneling**
4. Routing
5. Almacenamiento remoto
6. Configuración remota
7. Object Server

Para las conexiones Tunneling es suficiente con el Core y el Tunneling

3.1- KNXNET/IP Tunneling – Códigos

1. Servicios - Básicos (Core)

Table 4 – KNXnet/IP Core service type identifiers

Service name	Code	V.	Description
SEARCH_REQUEST	0201h	1	Sent by KNXnet/IP Client to search available KNXnet/IP Servers.
SEARCH_RESPONSE	0202h	1	Sent by KNXnet/IP Server when receiving a KNXnet/IP SEARCH_REQUEST.
DESCRIPTION_REQUEST	0203h	1	Sent by KNXnet/IP Client to a KNXnet/IP Server to retrieve information about capabilities and supported services.
DESCRIPTION_RESPONSE	0204h	1	Sent by KNXnet/IP Server in response to a DESCRIPTION_REQUEST to provide information about the server implementation.
CONNECT_REQUEST	0205h	1	Sent by KNXnet/IP Client for establishing a communication channel to a KNXnet/IP Server.
CONNECT_RESPONSE	0206h	1	Sent by KNXnet/IP Server as answer to CONNECT_REQUEST telegram.
CONNECTIONSTATE_REQUEST	0207h	1	Sent by KNXnet/IP Client for requesting the connection state of an established connection to a KNXnet/IP Server.
CONNECTIONSTATE_RESPONSE	0208h	1	Sent by KNXnet/IP Server when receiving a CONNECTIONSTATE_REQUEST for an established connection.
DISCONNECT_REQUEST	0209h	1	Sent by KNXnet/IP device, typically the KNXnet/IP Client, to terminate an established connection.
DISCONNECT_RESPONSE	020Ah	1	Sent by KNXnet/IP device, typically the KNXnet/IP Server, in response to a DISCONNECT_REQUEST.

3.1- KNXNET/IP Tunneling – Códigos

1. Servicios – Tunneling

Table 6 – Tunnelling KNXnet/IP service type identifiers

Service name	Code	V.	Description
TUNNELING_REQUEST	0420h	1	Used for sending and receiving single KNX telegrams between KNXnet/IP Client and - Server.
TUNNELING_ACK	0421h	1	Sent by a KNXnet/IP device to confirm the reception of the TUNNELING_REQUEST.

2. Errores - Connect_Response

Table 10 – Common CONNECT_RESPONSE status codes

Error constant	Value	V.	Description
E_NO_ERROR	00h	1	The connection is established successfully.
E_CONNECTION_TYPE	22h	1	The KNXnet/IP Server device does not support the requested connection type.
E_CONNECTION_OPTION	23h	1	The KNXnet/IP Server device does not support one or more requested connection options.
E_NO_MORE_CONNECTIONS	24h	1	The KNXnet/IP Server device cannot accept the new data connection because its maximum amount of concurrent connections is already used.

3.1- KNXNET/IP Tunneling – Códigos

2. Errores – ConnectionState_Response

Table 11 – CONNECTIONSTATE_RESPONSE status codes

Error constant	Value	V.	Description
E_NO_ERROR	00h	1	The connection state is normal.
E_CONNECTION_ID	21h	1	The KNXnet/IP Server device cannot find an active data connection with the specified ID.
E_DATA_CONNECTION	26h	1	The KNXnet/IP Server device detects an error concerning the data connection with the specified ID.
E_KNX_CONNECTION	27h	1	The KNXnet/IP Server device detects an error concerning the KNX connection with the specified ID.

2. Errores – Tunneling Connect_ACK

Table 12 – Tunnelling CONNECT_ACK error codes

Error constant	Value	V.	Description
E_NO_ERROR	00h	1	The message is received successfully.
E_TUNNELING_LAYER	29h	1	The KNXnet/IP Server device does not support the requested KNXnet/IP Tunnelling layer.

SERVICIOS TELEMÁTICOS AVANZADOS

CLASE 1 HASTA AQUÍ

3.1- KNXNET/IP Tunneling – Códigos

3. Protocolos compatibles

Table 16 – Host protocol codes for IP network

Constant name	Value	V.	Description
IPV4_UDP	01h	1	Identifies an Internet Protocol version 4 address and port number for UDP communication.
IPV4_TCP	02h	1	Identifies an Internet Protocol version 4 address and port number for TCP communication.

Table 18 – KNXnet/IP Internet Protocol constants

Description	Value	V.
KNXnet/IP Port Number	3671	1
KNXnet/IP System Setup Multicast Address	224.0.23.12	1

3.1- KNXNET/IP Tunneling – Códigos

4. Timeouts

Table 17 – Timeout constants

Constant name	Value	V.	Description
CONNECT_REQUEST_TIMEOUT	10 s	1	KNXnet/IP Client shall wait for 10 seconds for a CONNECT_RESPONSE frame from KNXnet/IP Server.
CONNECTIONSTATE_REQUEST_-TIMEOUT	10 s	1	KNXnet/IP Client shall wait for 10 seconds for a CONNECTIONSTATE_RESPONSE frame from KNXnet/IP Server.
DEVICE_CONFIGURATION_-REQUEST_TIMEOUT	10 s	1	KNXnet/IP Client shall wait for 10 seconds for a DEVICE_CONFIGURATION_RESPONSE frame from KNXnet/IP Server.
TUNNELING_REQUEST_TIMEOUT	1 s	1	KNXnet/IP Client shall wait for 1 second for a TUNNELING_ACK response on a TUNNELING_REQUEST frame from KNXnet/IP Server.
CONNECTION_ALIVE_TIME	120 s	1	If the KNXnet/IP Server does not receive a heartbeat request within 120 seconds of the last correctly received message frame, the server shall terminate the connection by sending a DISCONNECT_REQUEST to the client's control endpoint.

3.1- KNXNET/IP Tunneling – Discovery

Todas las aplicaciones software disponen de la capacidad de buscar los interfaces disponibles en la red LAN

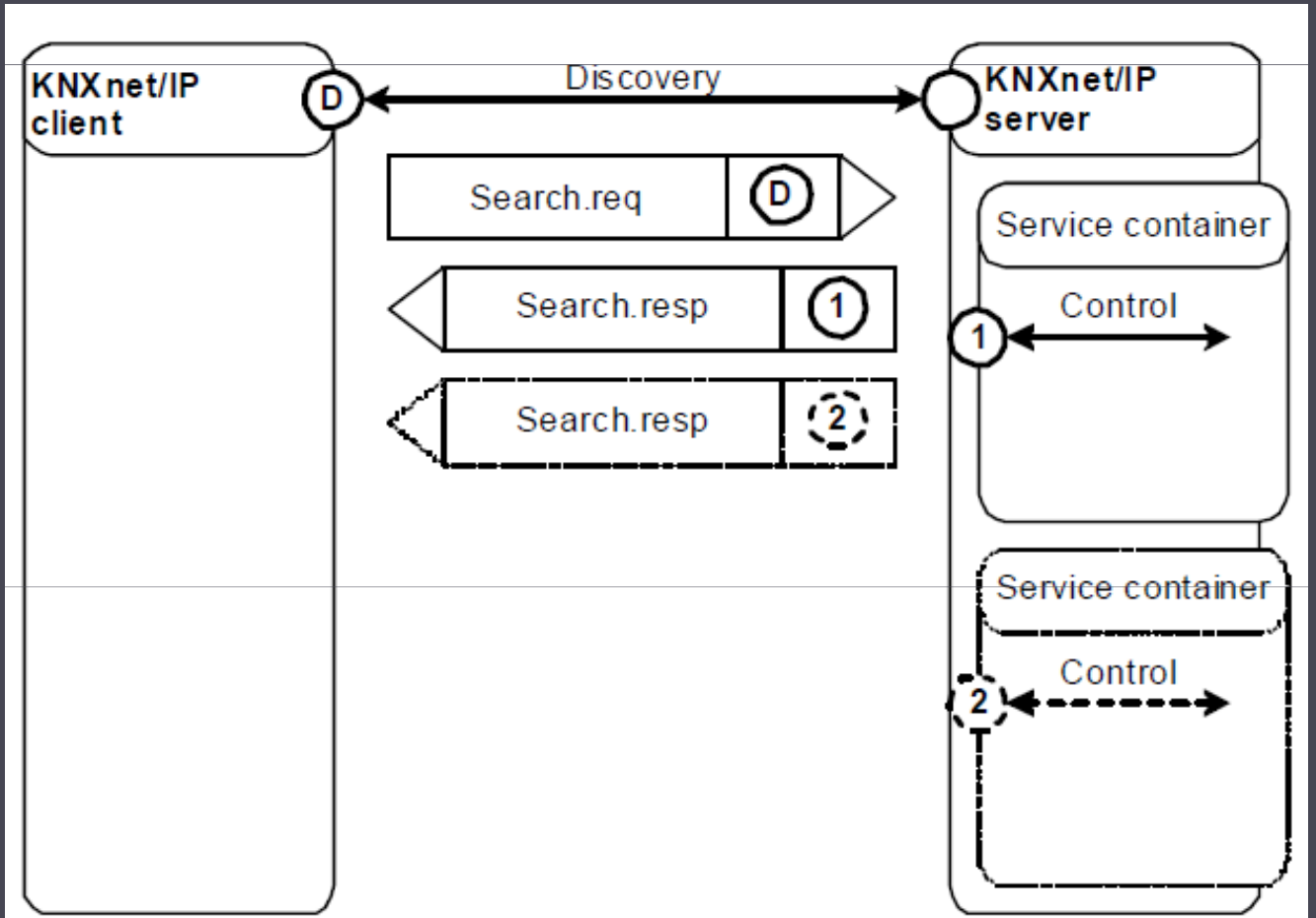
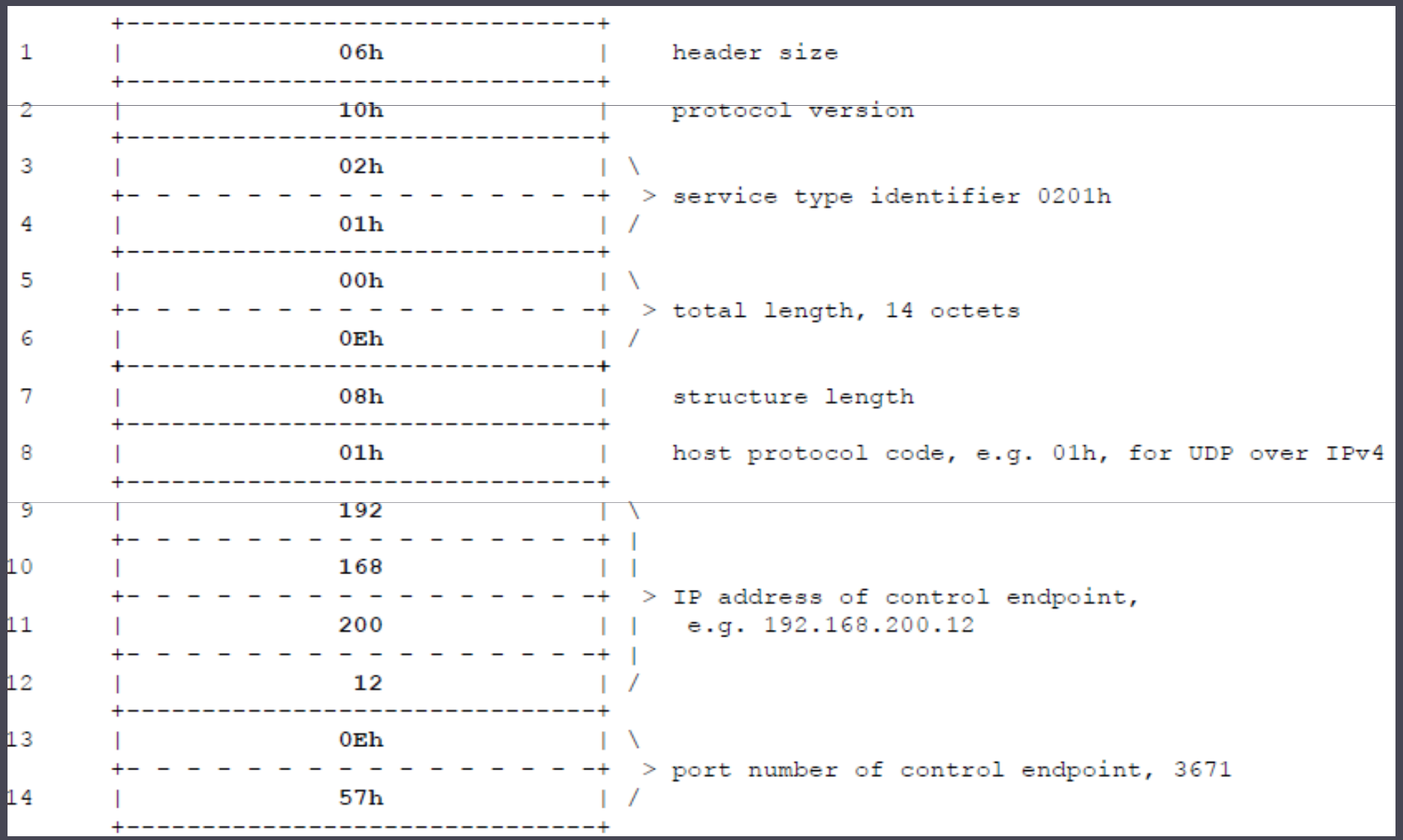


Figure 4 – Discovery procedure

3.1- KNXNET/IP Tunneling – Discovery

Ejemplo Search_request



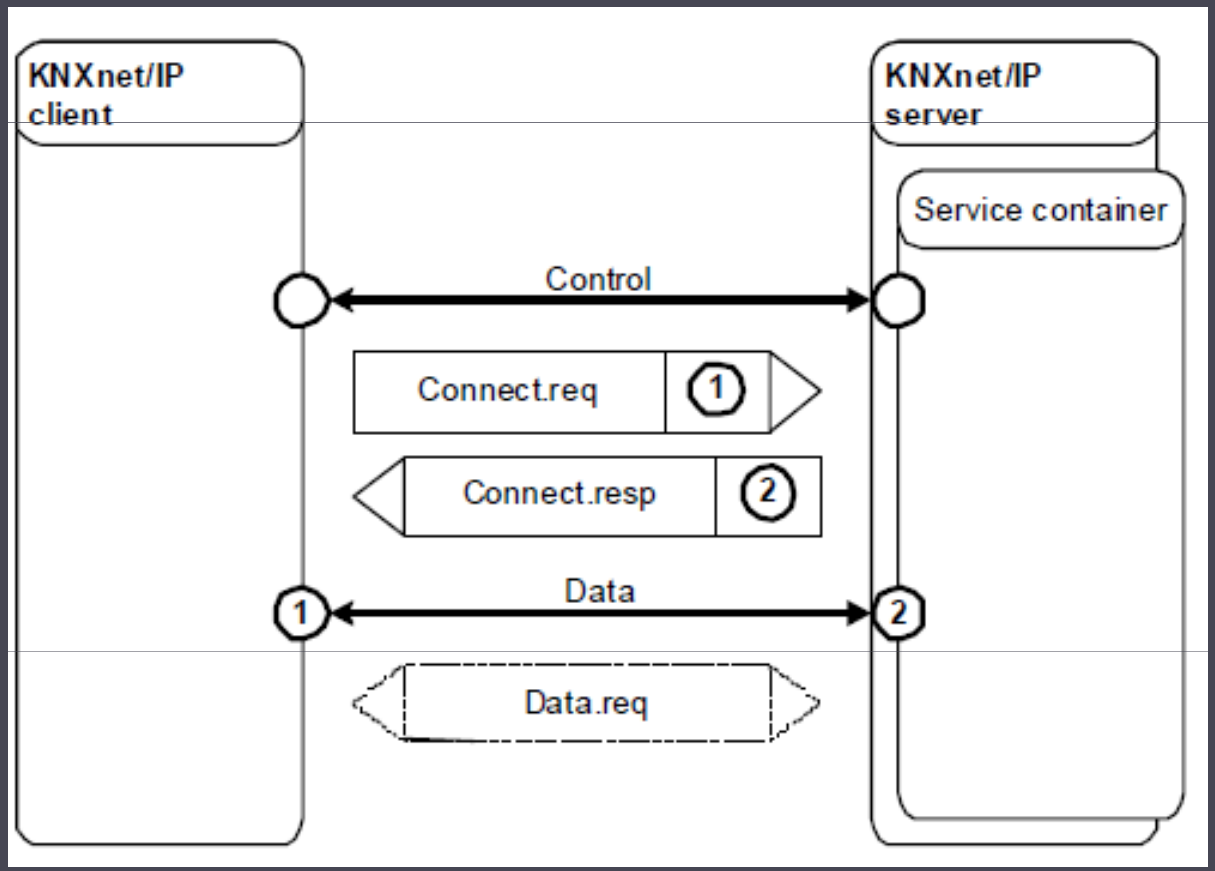
3.1- KNXNET/IP Tunneling – Discovery

Ejemplo
Search_response

1	06h	header size
2	10h	protocol version
3	02h	service type identifier 0202h
4	02h	
5	00h	total length, 78 octets
6	4Eh	
7	08h	structure length (HPAI)
8	01h	host protocol code, e.g. 01h, for UDP over
9	192	IP address of control endpoint, e.g. 192.168.200.12
10	168	
11	200	
12	12	
13	C3h	port number of control endpoint, e.g. 50100
14	B4h	
15	36h	structure length (DIB hardware)
16	01h	description type code, 01h = DEVICE_INFO
17	02h	KNX medium, 02h = TP1 (KNX TP)
18	01h	Device Status, 01h = programming mode
19	11h	KNX Individual Address, e.g. 1.1.0
20	00h	

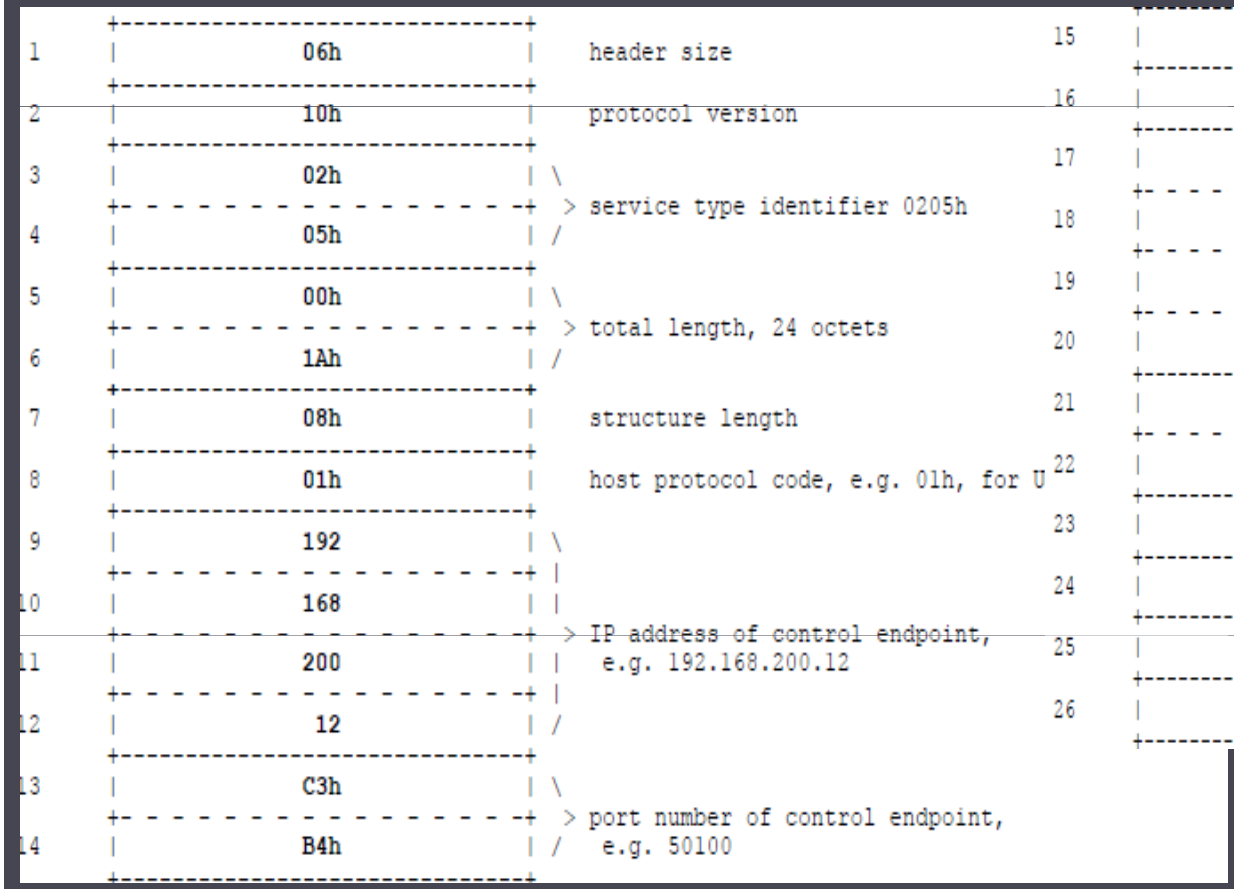
3.1- KNXNET/IP Tunneling – Conexión

Una vez descubierto el interfaz IP se puede realiza la conexión



3.1- KNXNET/IP Tunneling – Conexión

Ejemplo Connect_request



1	06h		header size
2	10h		protocol version
3	02h	\	
4	05h	/	> service type identifier 0205h
5	00h	\	
6	1Ah	/	> total length, 24 octets
7	08h		structure length
8	01h		host protocol code, e.g. 01h, for U
9	192	\	
10	168	|	
11	200	|	> IP address of control endpoint, e.g. 192.168.200.12
12	12	/	
13	C3h	\	
14	B4h	/	> port number of control endpoint, e.g. 50100
15	08h		structure length
16	01h		host protocol code, e.g. 01h, for UDP over IPv4
17	192	\	
18	168	|	
19	200	|	> IP address of data endpoint, e.g. 192.168.200.20
20	20	/	
21	C3h	\	
22	B4h	/	> port number of data endpoint, e.g. 50100
23	04h		structure length
24	04h		connection type code, e.g. 04h, TUNNEL_CONNECTION
25	02h		KNX layer, e.g. TUNNEL_LINKLAYER
26	00h		reserved

3.1- KNXNET/IP Tunneling – Conexión

Ejemplo Connect_response

1	06h	header size	
2	10h	protocol version	
3	02h	service type identifier 0206h	
4	06h		
5	00h	total length, 20 octets	
6	14h		
7	15h	communication channel ID, e.g. 21	
8	00h	status code (NO_ERROR)	
9	08h	structure length	
10	01h	host protocol code, e.g. 01h, for UDP over IPv4	
11	192	IP address of data endpoint, e.g. 192.168.200.20	
12	168		
13	200		
14	20		
15	C3h	port number of data endpoint, e.g. 50100	
16	B4h		
17	04h	structure length of CRD for TUNNELING_CONNECTION	
18	04h	connection type code, e.g. 04h, TUNNEL_CONNECTION	
19	11h	Individual Address, e.g. 01.01.10, used for TUNNELING_CONNECTION	
20	0Ah		

3.1- KNXNET/IP Tunneling – Conexión

Ejemplo ConnectionState_request

1	06h	header size
2	10h	protocol version
3	02h	service type identifier 0207h
4	07h	
5	00h	total length, 16 octets
6	10h	
7	15h	communication channel ID, e.g. 21
8	00h	reserved
9	08h	structure length
10	01h	host protocol code, e.g. 01h, for UDP
11	192	IP address of control endpoint, e.g. 192.168.200.12
12	168	
13	200	
14	12	
15	C3h	port number of control endpoint, e.g. 50100
16	B4h	

3.1- KNXNET/IP Tunneling – Conexión

Ejemplo ConnectionState_response

1	06h	header size
2	10h	protocol version
3	02h	service type identifier 0208h
4	08h	
5	00h	total length, 8 octets
6	08h	
7	15h	communication channel ID, e.g. 21
8	00h	status code (NO_ERROR)

3.1- KNXNET/IP Tunneling – Conexión

Ejemplo Disconnect_request

1	06h	header size
2	10h	protocol version
3	02h	service type identifier 0209h
4	09h	
5	00h	total length, 16 octets
6	10h	
7	15h	communication channel ID, e.g. 21
8	00h	reserved
9	08h	structure length
10	01h	host protocol code, e.g. 01h, for UDP
11	192	IP address of control endpoint, e.g. 192.168.200.12
12	168	
13	200	
14	12	
15	C3h	port number of control endpoint, e.g. 50100
16	B4h	

3.1- KNXNET/IP Tunneling – Conexión

Ejemplo Disconnect_response

1	06h	header size
2	10h	protocol version
3	02h	service type identifier 020Ah
4	0Ah	
5	00h	total length, 8 octets
6	08h	
7	15h	communication channel ID, e.g. 21
8	00h	status code (NO_ERROR)

3.1- KNXNET/IP Tunneling – Intercambio Tunneling

Ejemplo Tunneling_request

	KNXnet/IP header	
1	06h	header size
2	10h	protocol version
3	04h	service type identifier 0420h
4	20h	
5	00h	total length, L+12 octets
6	L+0Ch	
	connection header	
7	06h	structure length of connection header
8	15h	communication channel ID, e.g. 21
9	00h	sequence counter
10	00h	reserved
	cEMI frame	
11	11h	message code (e.g. L_Data.req message)
12	00h	additional information (none)
13	...	Service Information (L bytes)
14	...	
L+12	...	

Los mensajes cEMI son la información útil KNX (Telegramas IP)

3.1- KNXNET/IP Tunneling – Intercambio Tunneling

Ejemplo Tunneling_ACK

	- - - - **KNXnet/IP header** - - - -	
1	**06h**	header size
2	**10h**	protocol version
3	**04h**	\
4	**21h**	> service type identifier 0421h /
5	**00h**	\
6	**0Ah**	> total length, 10 octets /
	- - - - **connection header** - - - -	
7	**04h**	structure length of connection header
8	**15h**	communication channel ID, e.g. 21
9	**00h**	sequence counter
10	**00h**	status, e.g. 00h (NO_ERROR)

3.1- KNXNET/IP Tunneling – Mensajes cEMI

Este tipo de mensaje es el utilizado para conexiones independientes del medio

Message Code	Additional Info Length	Additional Information	Control field 1	Control field 2	Src. High	Src. Low	Dest. High	Dest. Low	NPDU	
MC	**AddIL**	…	**Ctrl1**	**Ctrl2**	**SAH**	**SAL**	**DAH**	**DAL**	**L**	TPCI/APCI & data
1 octet	1 octet	var. length	1 octet	1 octet	2 octets		2 octets		1 octet	var. length

Message Code	
L_Busmon.ind	2Bh
L_Data.req	11h
L_Data.con	2Eh
L_Data.ind	29h

Al Bus

Confirmación (Contesta a una lectura anterior)

Desde el bus

3.1- KNXNET/IP Tunneling – Mensajes cEMI

Este tipo de mensaje es el utilizado para conexiones independientes del medio

AddIL= 00h no Add Info

Message Code	Additional Info Length	Additional Information	Control field 1	Control field 2	Src. High	Src. Low	Dest. High	Dest. Low	NPDU	
MC	**AddIL**	…	**Ctrl1**	**Ctrl2**	**SAH**	**SAL**	**DAH**	**DAL**	**L**	TPCI/APCI & data
1 octet	1 octet	var. length	1 octet	1 octet	2 octets		2 octets		1 octet	var. length

1 octet							
Ctrl1							
7	6	5	4	3	2	1	0
FT	0	R	SB	P		A	C

Priority (P) (bit 3 and bit 2): Priority that shall be used for transmission

Confirm flag (C) Error in the transmitted frame

1 octet							
Ctrl2							
7	6	5	4	3	2	1	0
AT	HC			EFF			

Destination Address Type (AT) 0: individual / 1: group

Hop Count (HC) (bit 6 to bit 4) Contador de ruta

3.1- KNXNET/IP Tunneling – Mensajes cEMI

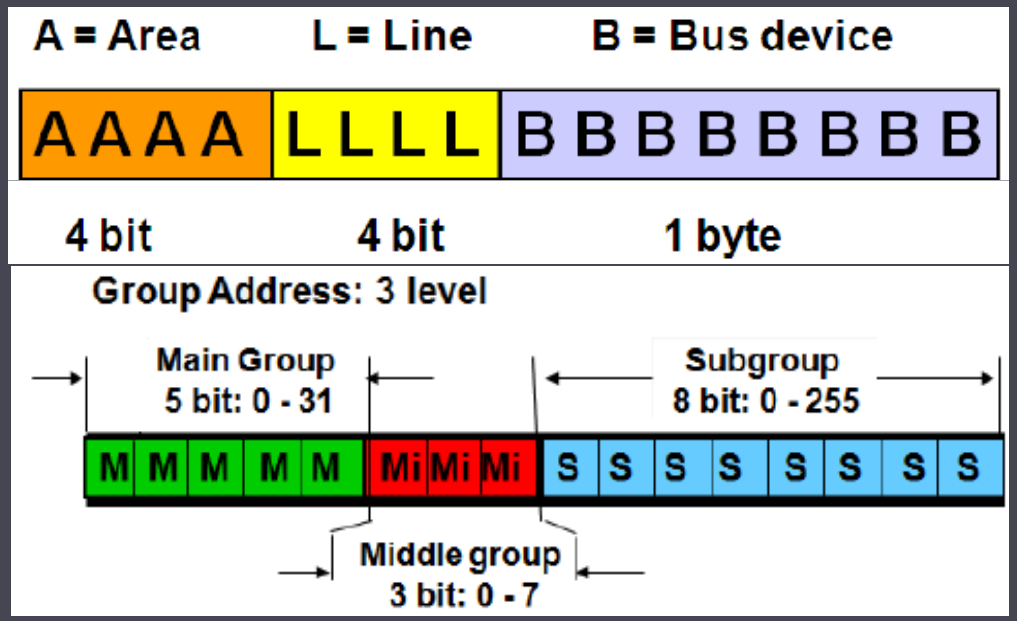
Este tipo de mensaje es el utilizado para conexiones independientes del medio

00h no Add Info

Message Code	Additional Info Length	Additional Information	Control field 1	Control field 2	Src. High	Src. Low	Dest. High	Dest. Low	NPDU	
MC	**AddIL**	**…**	**Ctrl1**	**Ctrl2**	**SAH**	**SAL**	**DAH**	**DAL**	**L**	TPCI/APCI & data
1 octet	1 octet	var. length	1 octet	1 octet	2 octets		2 octets		1 octet	var. length

Dirección Física

Dirección de grupo



3.1- KNXNET/IP Tunneling – Mensajes cEMI

Este tipo de mensaje es el utilizado para conexiones independientes del medio

Message Code	Additional Info Length	Additional Information	Control field 1	Control field 2	Src. High	Src. Low	Dest. High	Dest. Low	NPDU	
MC	**AddIL**	…	**Ctrl1**	**Ctrl2**	**SAH**	**SAL**	**DAH**	**DAL**	**L**	TPCI/APCI & data
1 octet	1 octet	var. length	1 octet	1 octet	2 octets		2 octets		1 octet	var. length

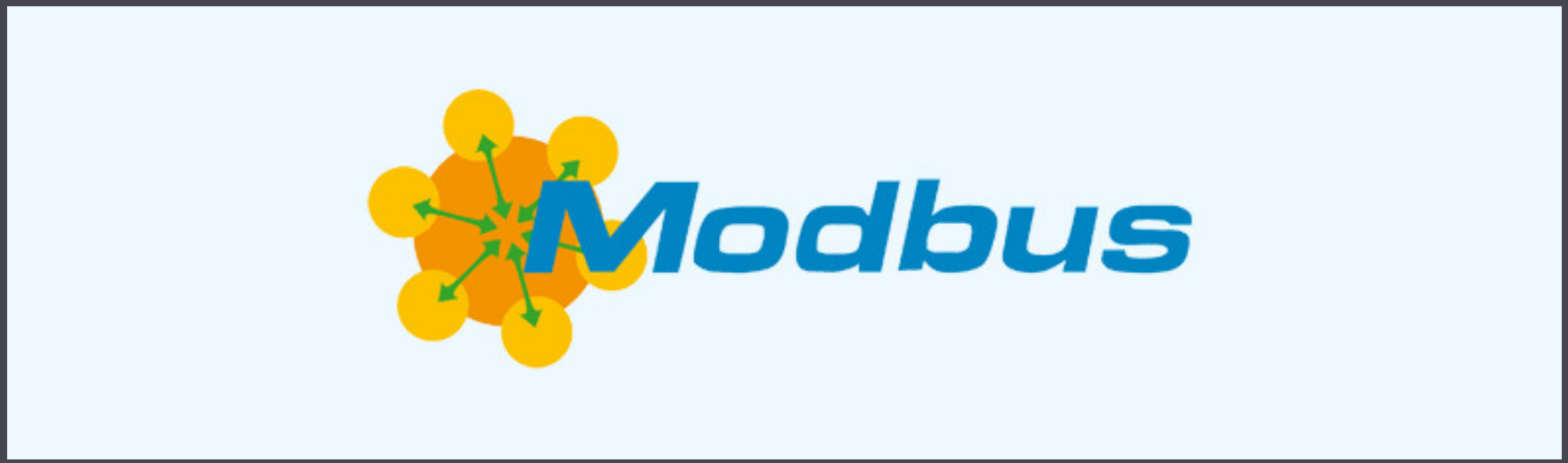
L: Information-Length en bytes (max. value is 255)

7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	
						APCI	APCI	APCI/dat	APCI/dat	APCI/dat	APCI/dat	APCI/dat	APCI/dat	APCI/dat	APCI/dat	
						0	0	0	0	0	0	0	0	0	0	A_GroupValue_Read-PDU
						0	0	0	1							A_GroupValue_Response-PDU
						0	0	1	0							A_GroupValue_Write-PDU

3.2- Modbus TCP

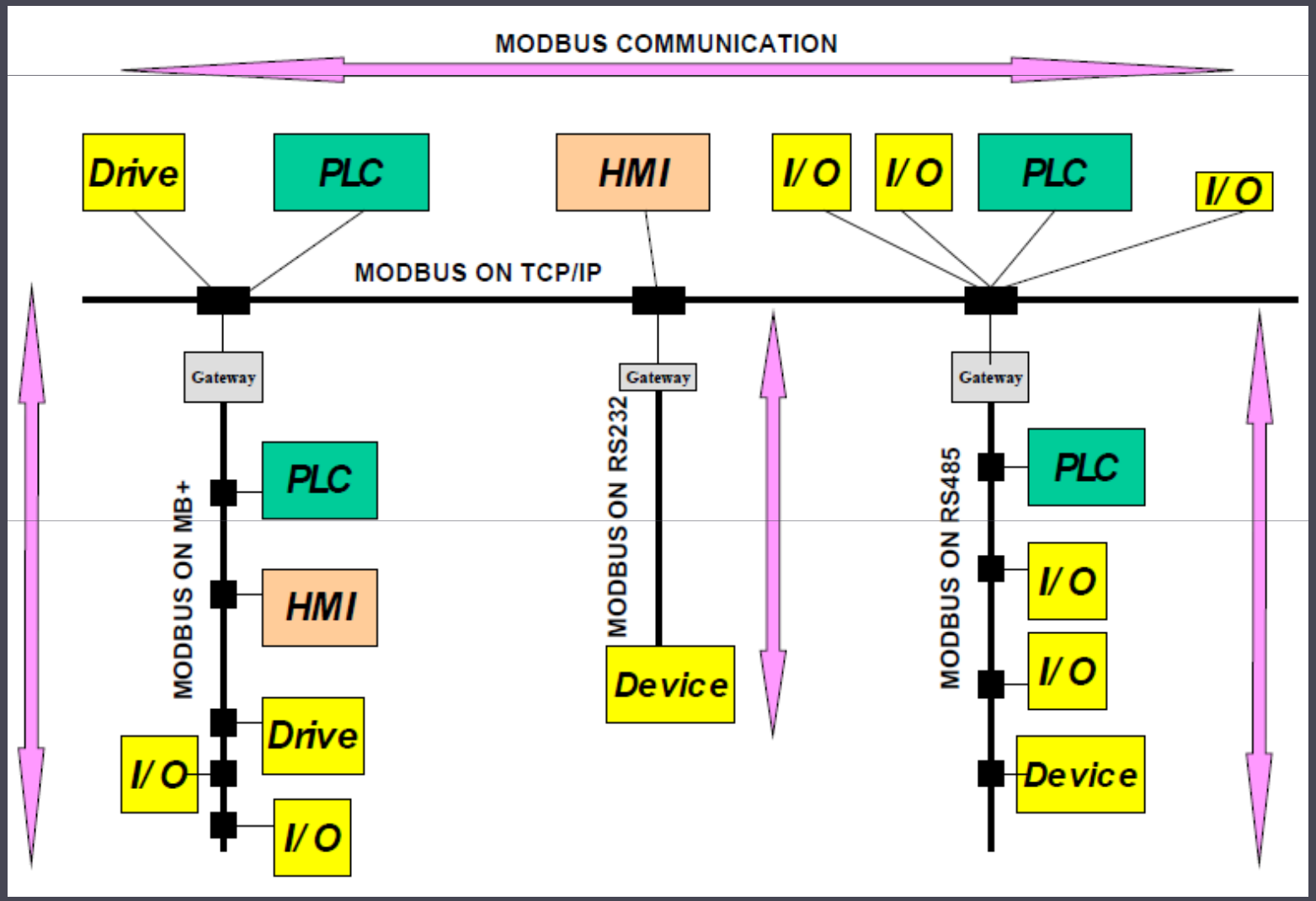
MODBUS is an application layer messaging protocol, positioned at level 7 of the OSI model, which provides client/server communication between devices connected on different types of buses or networks.

The industry's serial de facto standard since 1979, MODBUS continues to enable millions of automation devices to communicate.



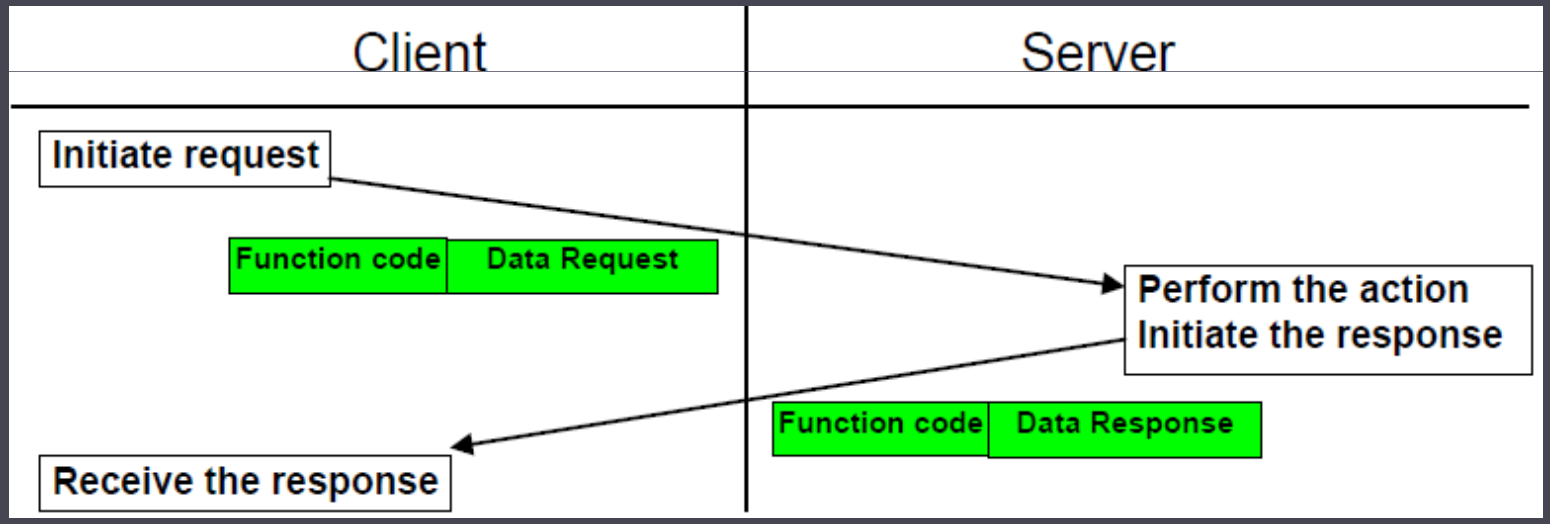
3.2- Modbus TCP

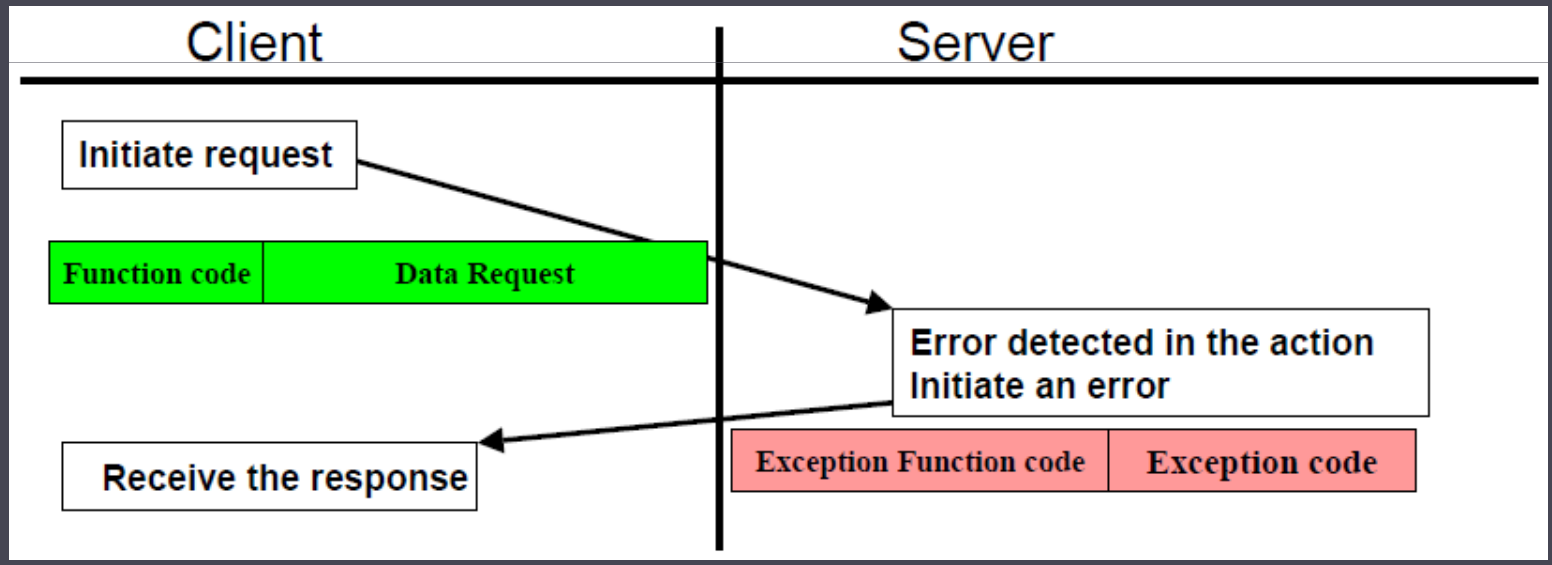
En Modbus TCP confluyen todas sus tecnologías



3.2- Modbus TCP

No existe establecimiento de la conexión





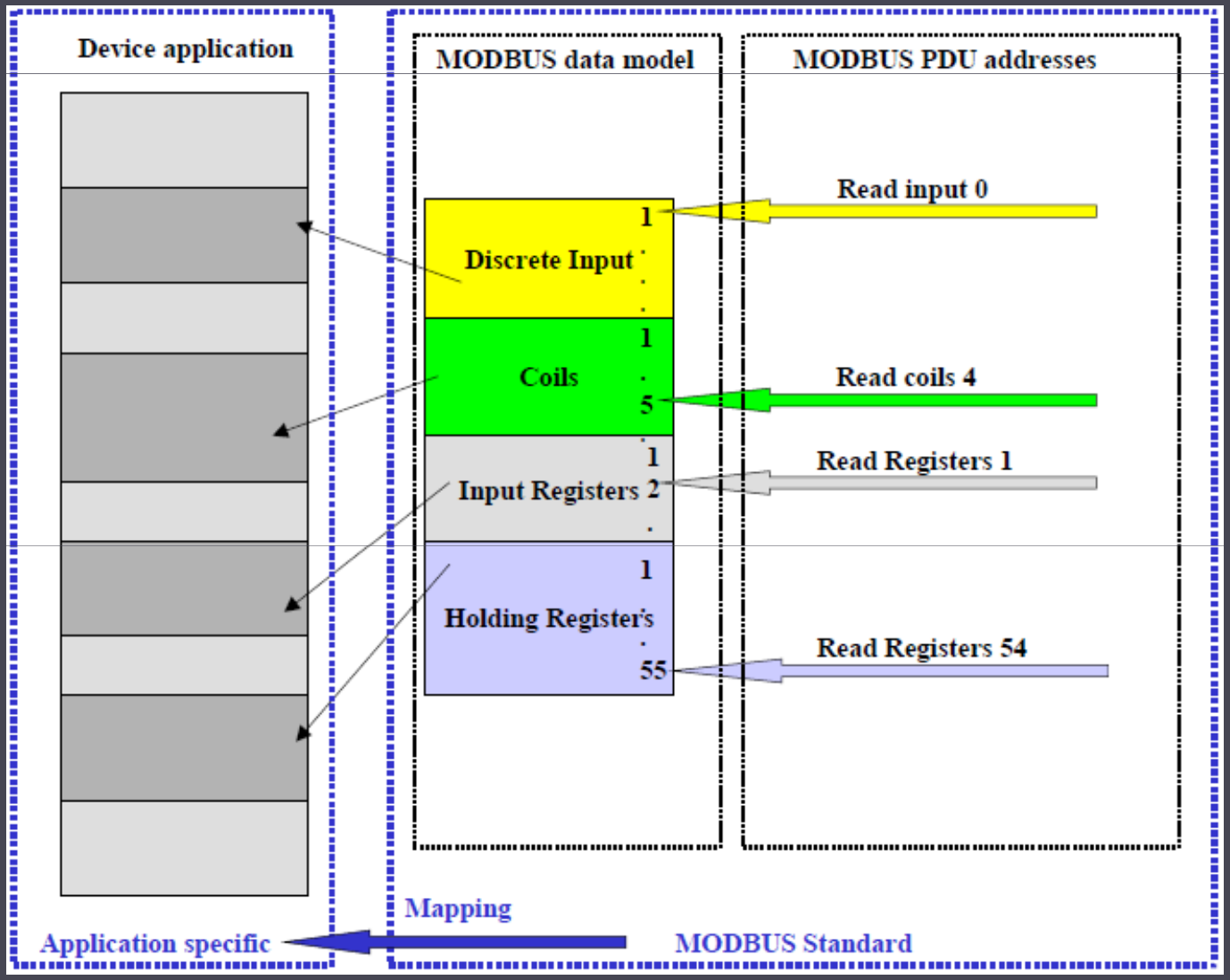
3.2- Modbus TCP

Los tipos de datos disponibles son los siguientes:

Primary tables	Object type	Type of	Comments
Discretes Input	Single bit	Read-Only	This type of data can be provided by an I/O system.
Coils	Single bit	Read-Write	This type of data can be alterable by an application program.
Input Registers	16-bit word	Read-Only	This type of data can be provided by an I/O system
Holding Registers	16-bit word	Read-Write	This type of data can be alterable by an application program.

3.2- Modbus TCP

Mapas de memoria en dispositivos Modbus



Dirección 0 -> Registro 1

Dirección 5 -> Registro 5

…

3.2- Modbus TCP

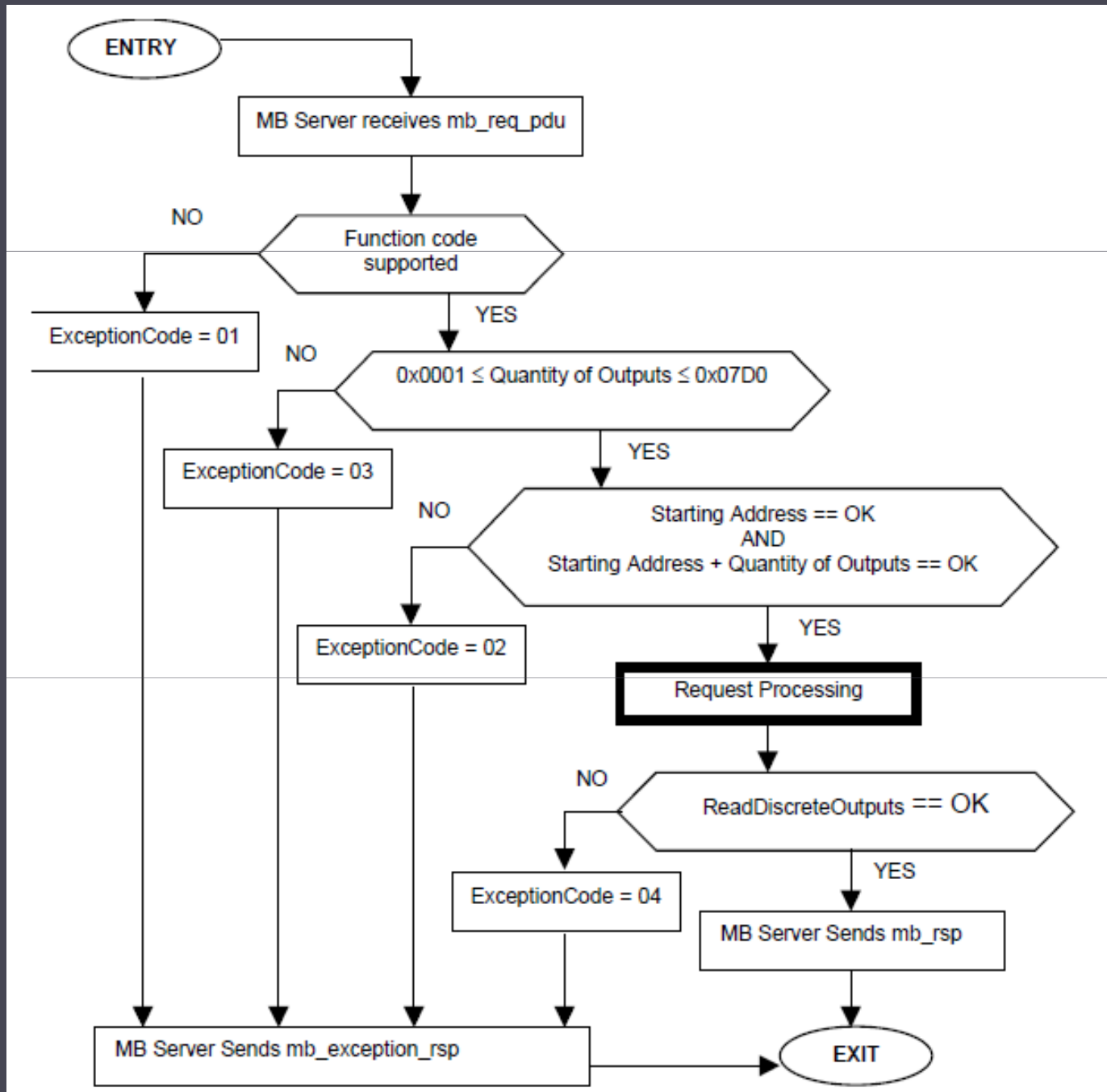
Funciones disponibles de intercambio de datos

				Function Codes		
				code	*Sub code*	*(hex)*
Data Access	**Bit access**	Physical Discrete Inputs	Read Discrete Inputs	**02**		**02**
		Internal Bits Or Physical coils	Read Coils	**01**		**01**
			Write Single Coil	**05**		**05**
			Write Multiple Coils	**15**		**0F**
	16 bits access	Physical Input Registers	Read Input Register	**04**		**04**
		Internal Registers Or Physical Output Registers	Read Holding Registers	**03**		**03**
			Write Single Register	**06**		**06**
			Write Multiple Registers	**16**		**10**
			Read/Write Multiple Registers	**23**		**17**
			Mask Write Register	**22**		**16**
			Read FIFO queue	**24**		**18**

Es posible solicitar/modificar múltiples registros en un único request

3.2- Modbus TCP

Proceso recepción solicitud



3.2- Modbus TCP

Ejemplo solicitud de lectura de los Coils 20-38

Request		Response	
Field Name	*(Hex)*	*Field Name*	*(Hex)*
Function	**01**	Function	**01**
Starting Address Hi	**00**	Byte Count	**03**
Starting Address Lo	**13**	Outputs status 27-20	**CD**
Quantity of Outputs Hi	**00**	Outputs status 35-28	**6B**
Quantity of Outputs Lo	**13**	Outputs status 38-36	**05**

DATOS en formato big-Endian

	27-20	35-28	38-36
hex	CD	6B	05
bin	1100 1101	0110 1011	0000 0101

3.2- Modbus TCP

Ejemplo solicitud de lectura de los Holding register 108-110

Request		Response	
Field Name	*(Hex)*	*Field Name*	*(Hex)*
Function	03	Function	03
Starting Address Hi	00	Byte Count	06
Starting Address Lo	6B	Register value Hi (108)	02
No. of Registers Hi	00	Register value Lo (108)	2B
No. of Registers Lo	03	Register value Hi (109)	00
		Register value Lo (109)	00
		Register value Hi (110)	00
		Register value Lo (110)	64

3.2- Modbus TCP

Modbus header

	Description	Size	Example
MBAP Header	Transaction Identifier Hi	1	0x15
	Transaction Identifier Lo	1	0x01
	Protocol Identifier	2	0x0000
	Length	2	0x0006
	Unit Identifier	1	0xFF
MODBUS request	*Function Code (*)*	*1*	*0x03*
	Starting Address	*2*	*0x0004*
	Quantity of Registers	*2*	*0x0001*

Modbus

Transaction Identifier: Es un contador que permite enlazar los request con los response.

Unit Identifier: Es necesario para el direccionamiento de equipos esclavos.

3.2- Modbus TCP

Respuesta a errores

Request

Function code	1 Byte	**0x08**
Sub-function	2 Bytes	
Data	N x 2 Bytes	

Response

Function code	1 Byte	**0x08**
Sub-function	2 Bytes	
Data	N x 2 Bytes	

OK

Error

Error code	1 Byte	**0x88**
Exception code	1 Byte	01 or 03 or 04

ERROR

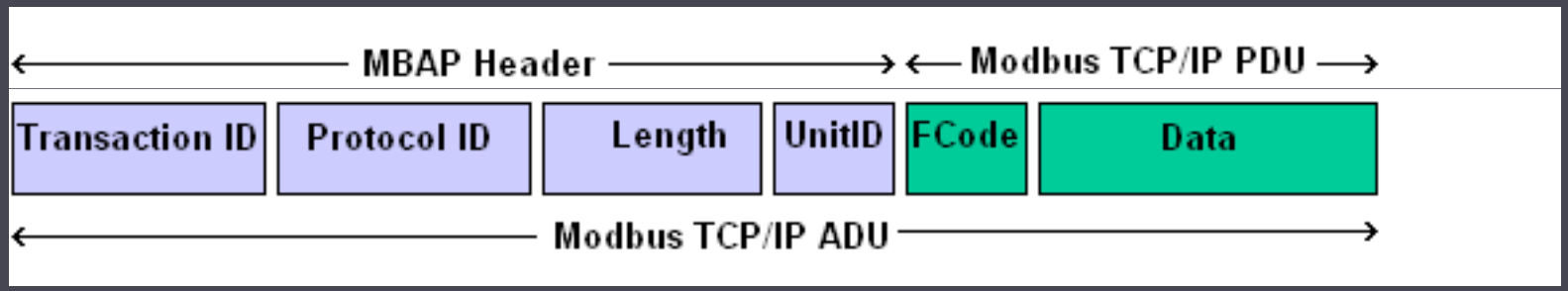
3.2- Modbus TCP

Exception codes

Exception Code	MODBUS name	Comments
01	Illegal Function Code	The function code is unknown by the server
02	Illegal Data Address	Dependant on the request
03	Illegal Data Value	Dependant on the request
04	Server Failure	The server failed during the execution
05	Acknowledge	The server accepted the service invocation but the service requires a relatively long time to execute. The server therefore returns only an acknowledgement of the service invocation receipt.
06	Server Busy	The server was unable to accept the MB Request PDU. The client application has the responsibility of deciding if and when to re-send the request.
0A	Gateway problem	Gateway paths not available.
0B	Gateway problem	The targeted device failed to respond. The gateway generates this exception

3.2- Modbus TCP

Ejemplo telegrama Modbus

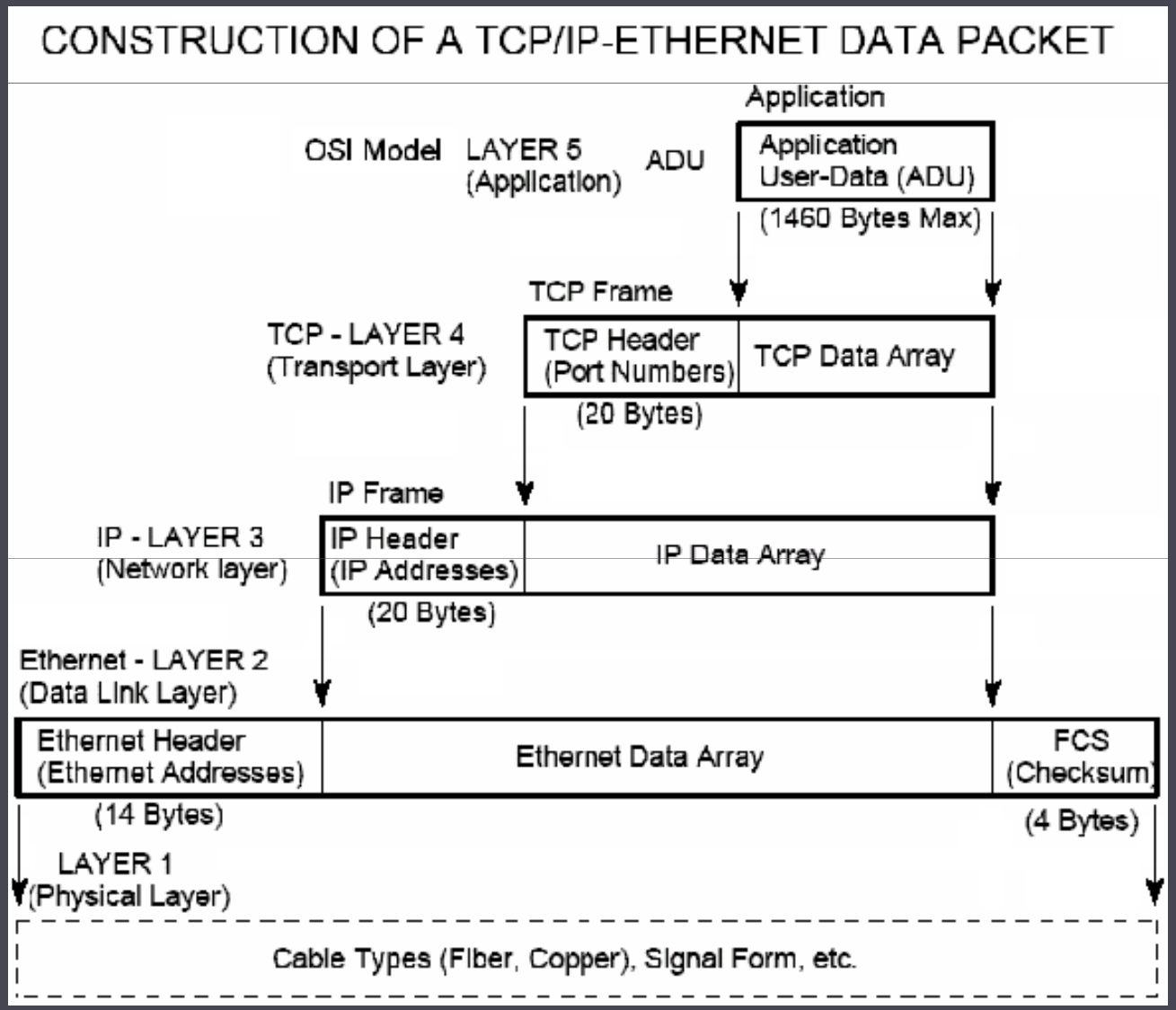


0001 0000 0006 11 03 006B 0003

0001: Transaction Identifier
0000: Protocol Identifier
0006: Message Length (6 bytes to follow)
11: The Unit Identifier (17 = 11 hex)
03: The Function Code (read Analog Output Holding Registers)
006B: The Data Address of the first register requested. (107 =6B hex)
0003: The total number of registers requested. (read 3 registers 107 to 109)

3.2- Modbus TCP

Ejemplo telegrama Modbus



Modbus ADU

3.3- BACnet/IP

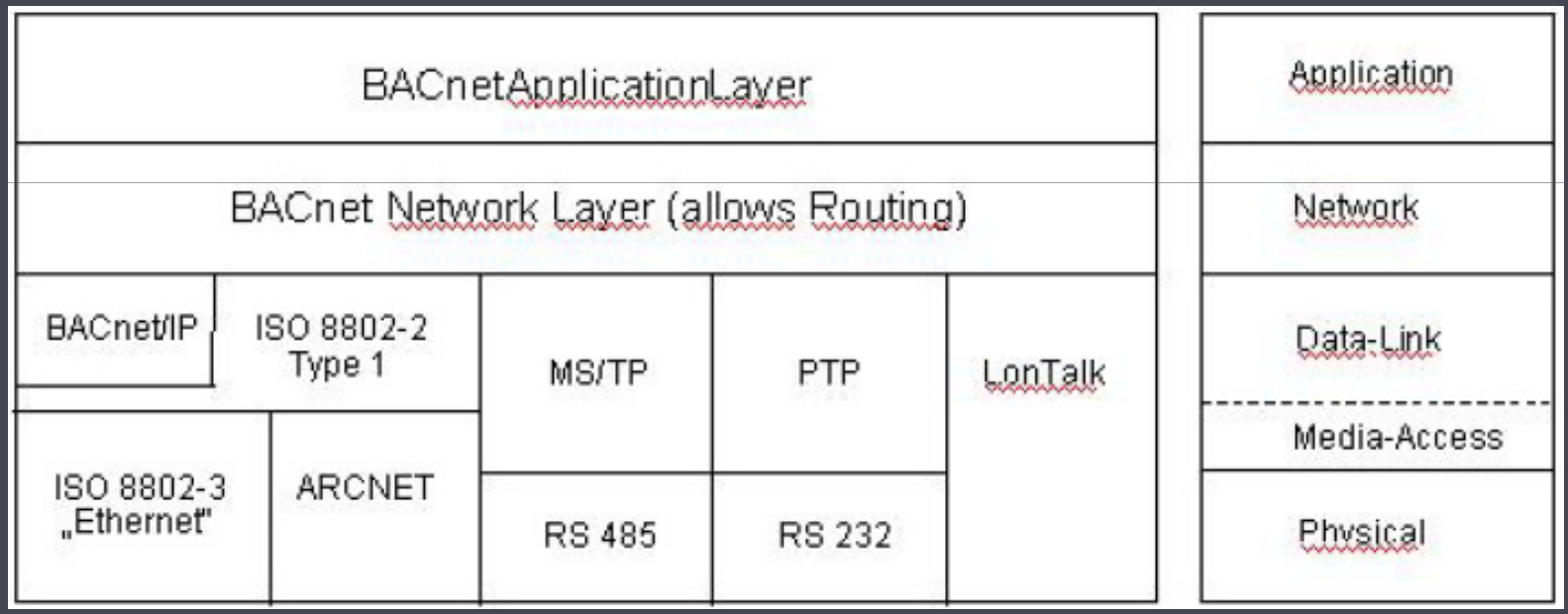
Building Automation and Controls NETwork

Developed under the auspices of the American Society of Heating, Refrigerating and Air-Conditioning Engineers (ASHRAE), BACnet is an American national standard, a European standard, a national standard in more than 30 countries, and an ISO global standard.

3.3- BACnet/IP

Las comunicaciones en BACnet utilizan diferentes medios físicos:

- **Ethernet**
- **BACnet/IP**
- **Serie (RS232/RS485)**
- **ARCnet**
- **MS/TP**
- **LonTalk**



3.3- BACnet/IP

La trama de BACnet tiene la siguiente estructura:

0x55	Preamble	2 octets
0xFF		
Frame Type		1 octet
Destination Address		1 octet
Source Address		1 octet
Length		2 octets, most significant octet first
Header CRC		1 octet
Data		(present if Length is non-zero) Length octets
Data CRC		(present if Length is non-zero) 2 octets
0xFF	pad	(optional) at most, 1 octet

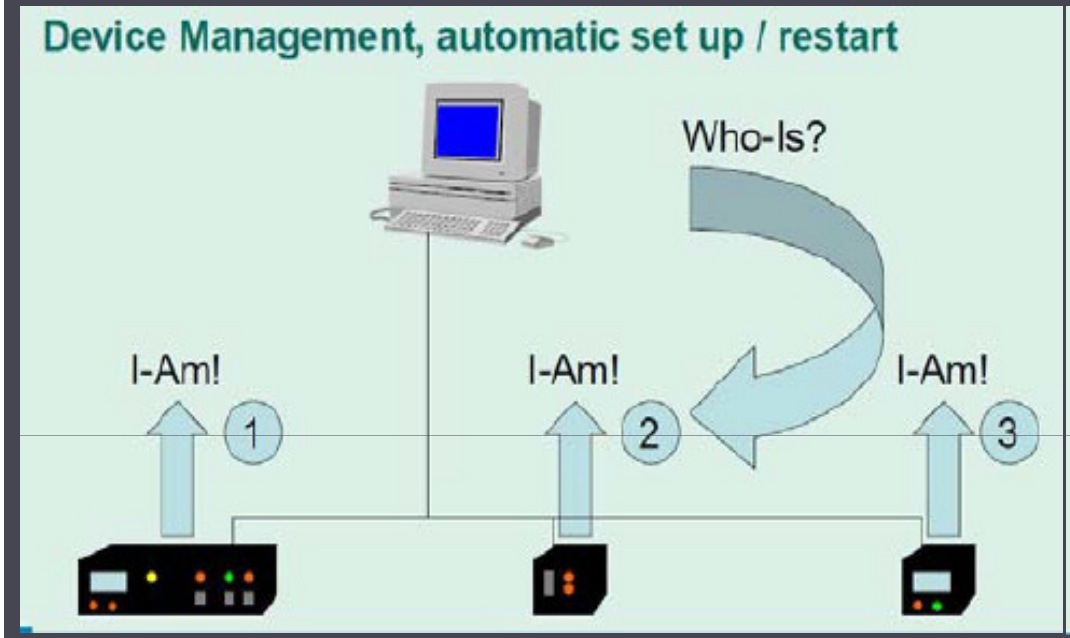
3.3- BACnet/IP

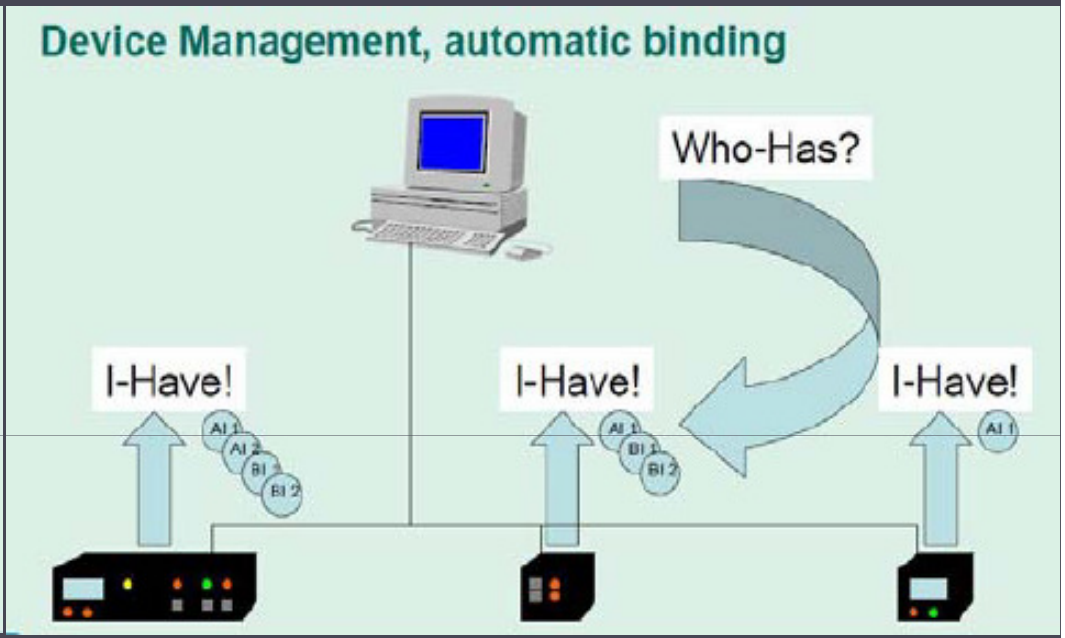
Áreas de interoperabilidad de los servicios

1. Data Sharing DS (Intercambio de datos)
2. Alarm and Event Management AE (Distribución de notificaciones y Ack. de alarmas)
3. Scheduling SCHED (Acciones que dependen de horarios y calendarios)
4. Trending T (Registro de variables y eventos)
5. Device and Network Management DM, NM, VT

3.3- BACnet/IP

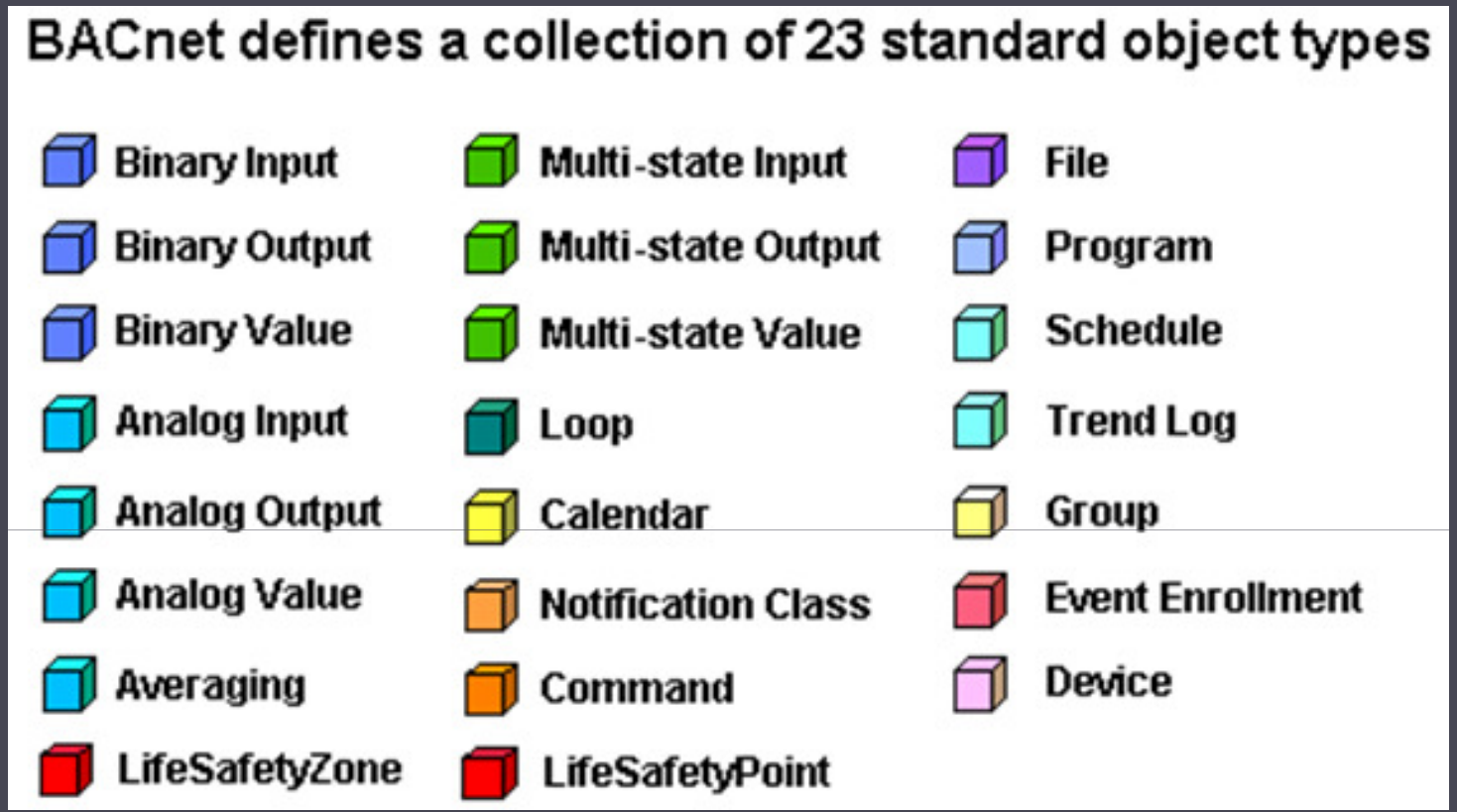
Ejemplos de servicios Device Management





3.3- BACnet/IP

BACnet crea objetos con múltiples valores y propiedades



3.3- BACnet/IP

Las propiedades de los objetos son obligatorias (estandarización) u opcionales

Las propiedades pueden ser de lectura y/o escritura

Object_Name	ROOM_TEMP
Object_Type	ANALOG INPUT
Present_Value	20.3
Unit	62 = °C
High_Limit	30.0
Low_Limit	15.0

object-identifier	[75]	BACnetObjectIdentifier,
object-name	[77]	CharacterString,
object-type	[79]	BACnetObjectType,
present-value	[85]	REAL,
description	[28]	CharacterString OPTIONAL,
device-type	[31]	CharacterString OPTIONAL,
status-flags	[111]	BACnetStatusFlags,
event-state	[36]	BACnetEventState,
reliability	[103]	BACnetReliability OPTIONAL,
out-of-service	[81]	BOOLEAN,
update-interval	[118]	Unsigned OPTIONAL,
units	[117]	BACnetEngineeringUnits,
min-pres-value	[69]	REAL OPTIONAL,
max-pres-value	[65]	REAL OPTIONAL,
resolution	[106]	REAL OPTIONAL
cov-increment	[22]	REAL OPTIONAL,
time-delay	[113]	Unsigned OPTIONAL,
notification-class	[17]	Unsigned OPTIONAL,
high-limit	[45]	REAL OPTIONAL,

3.3- BACnet/IP

Ejemplo objeto de actuador

Property Identifier	Property Datatype	Conformance Code
~~Object_Identifier~~	~~BACnetObjectIdentifier~~	R
Object_Name	CharacterString	R
Object_Type	BACnetObjectType	R
Present_Value	BACnetBinaryPV	W
Description	CharacterString	O
Device_Type	CharacterString	O
Status_Flags	BACnetStatusFlags	R
Event_State	BACnetEventState	R
Reliability	BACnetReliability	O
Out_Of_Service	BOOLEAN	R
Polarity	BACnetPolarity	R
Inactive_Text	CharacterString	O[1]
Active_Text	CharacterString	O[1]
Change_Of_State_Time	BACnetDateTime	O[2]
Change_Of_State_Count	Unsigned	O[2]
Time_Of_State_Count_Reset	BACnetDateTime	O[2]
Elapsed_Active_Time	Unsigned32	O[3]
Time_Of_Active_Time_Reset	BACnetDateTime	O[3]
Minimum_Off_Time	Unsigned32	O
Minimum_On_Time	Unsigned32	O
Priority_Array	BACnetPriorityArray	R
Relinquish_Default	BACnetBinaryPV	R
Time_Delay	Unsigned	O[4]
Notification_Class	Unsigned	O[4]
Feedback_Value	BACnetBinaryPV	O[4]
Event_Enable	BACnetEventTransitionBits	O[4]
Acked_Transitions	BACnetEventTransitionBits	O[4]
Notify_Type	BACnetNotifyType	O[4]
Event_Time_Stamps	BACnetARRAY[3] of BACnetTimeStamp	O[4]
Profile_Name	CharacterString	O

Conformance Codes:
R required, readable
W required, writable
O optional

Out_Of_Service decouples the physical output from the Present_Value.

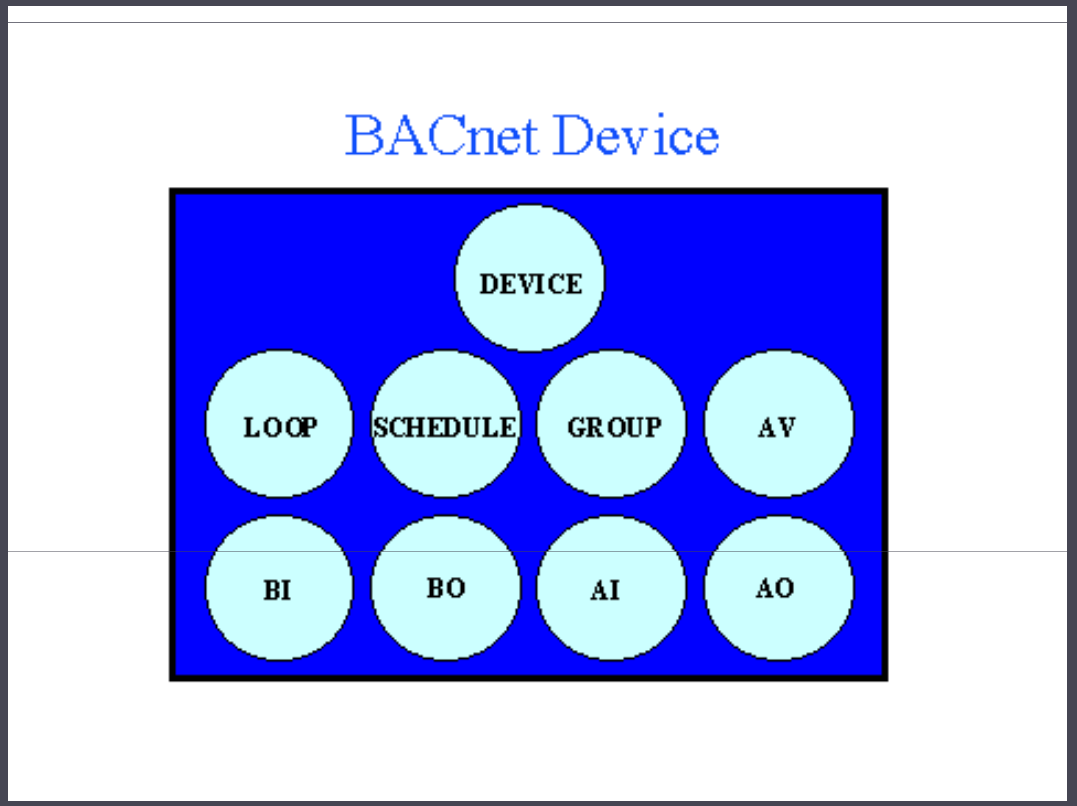
Properties required because Present_Value is *commandable*

Properties required for intrinsic reporting

11

3.3- BACnet/IP

Un dispositivo BACnet está compuesto por múltiples objetos



3.3- BACnet/IP

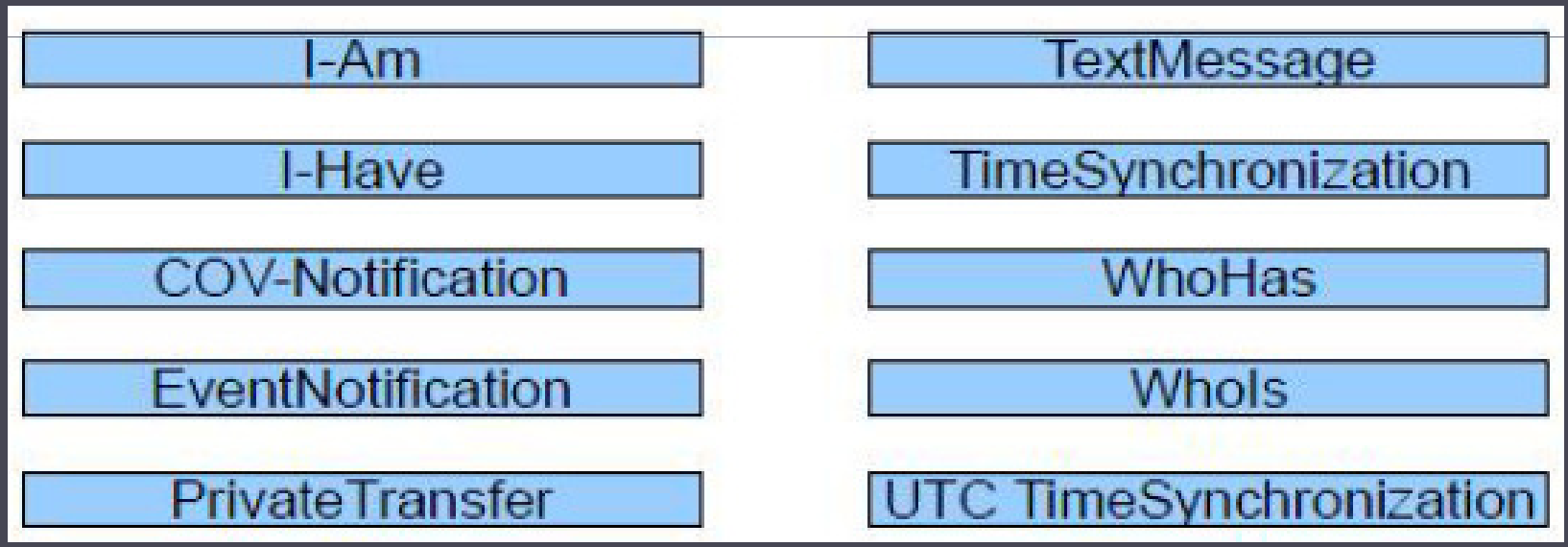
El acceso a los objetos se realiza a través de los "Object Access Services"

Estos servicios son Unicast

AcknowledgeAlarm	AtomicWriteFile	WritePropertyMultiple
COV-Notification	AddListElement	PrivateTransfer
EventNotification	RemoveListElement	TextMessage
GetAlarmSummary	CreateObject	ReinitializeDevice
GetEnrollmentSummary	DeleteObject	VTOpen
SubscribeCOV	ReadProperty	VTClose
SubscribeCOVProperty	ReadPropertyMultiple	VTData
LifeSafetyOperation	ReadRange	Authenticate
AtomicReadFile	WriteProperty	RegestKey
ReadPropertyConditional		DeviceCommunicationControl

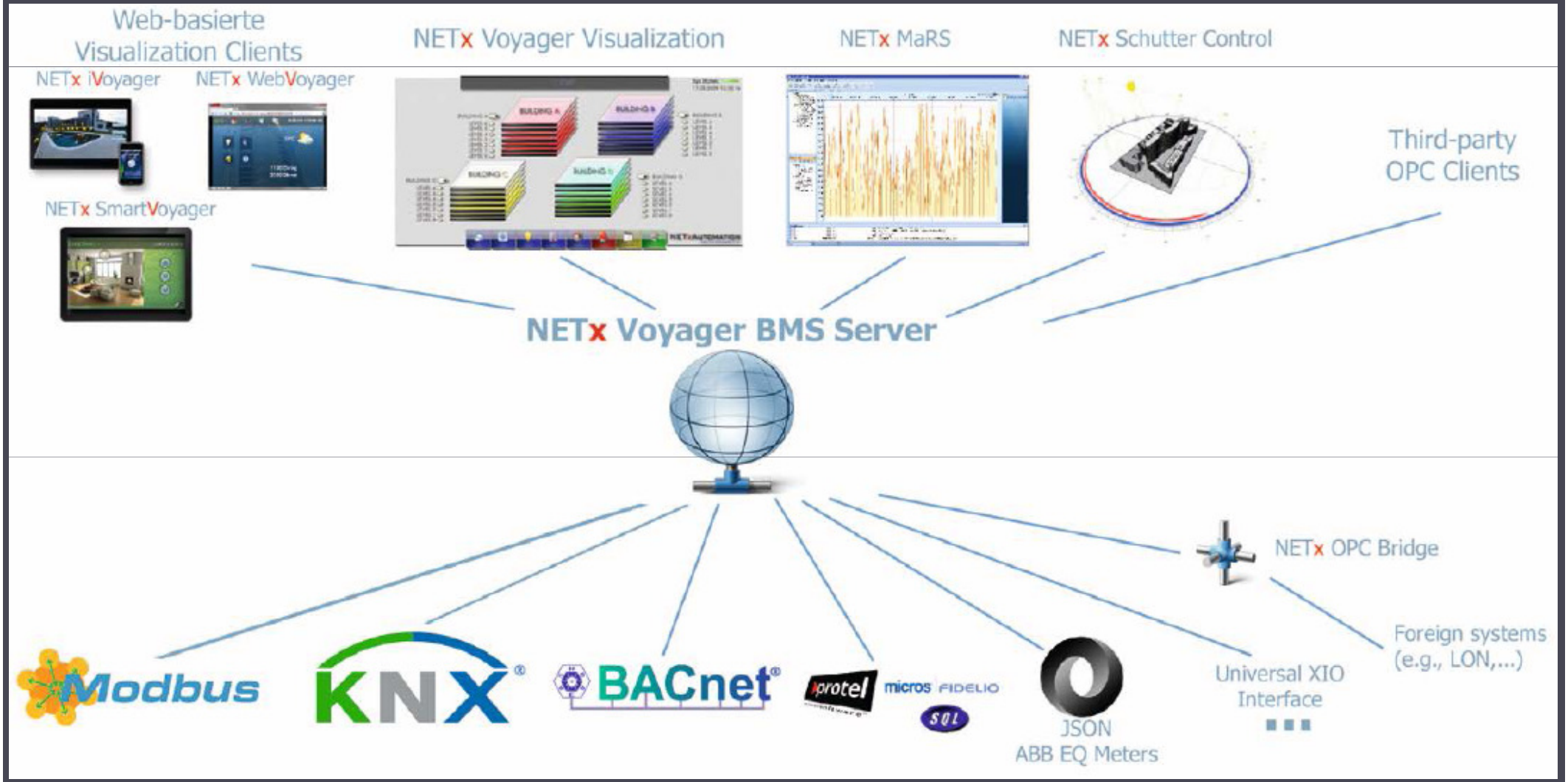
3.3- BACnet/IP

Estos servicios comunes a varios dispositivos son Multicast



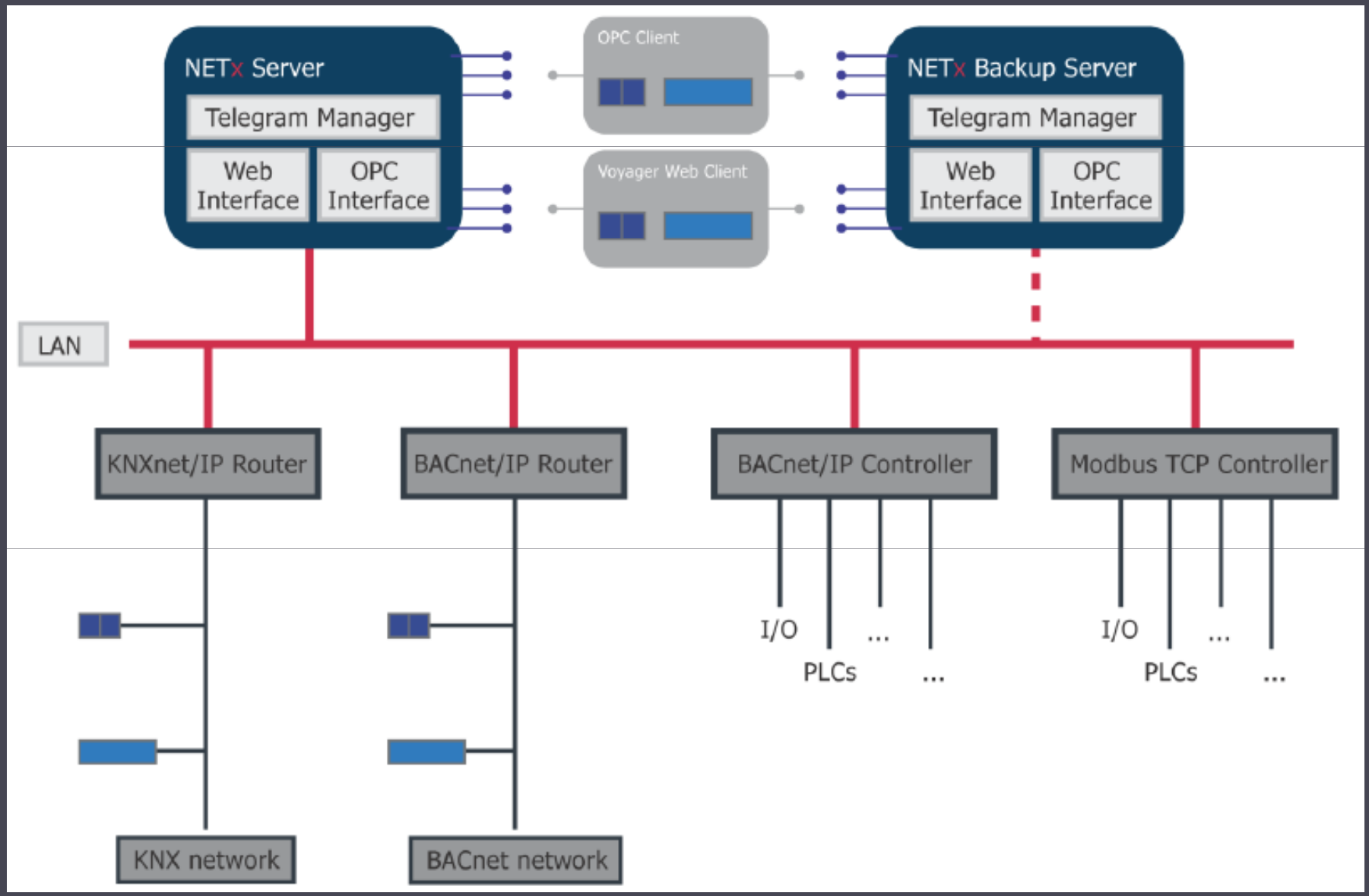
SERVICIOS TELEMÁTICOS AVANZADOS

4- EJEMPLO BMS MULTIPROTOCOLO



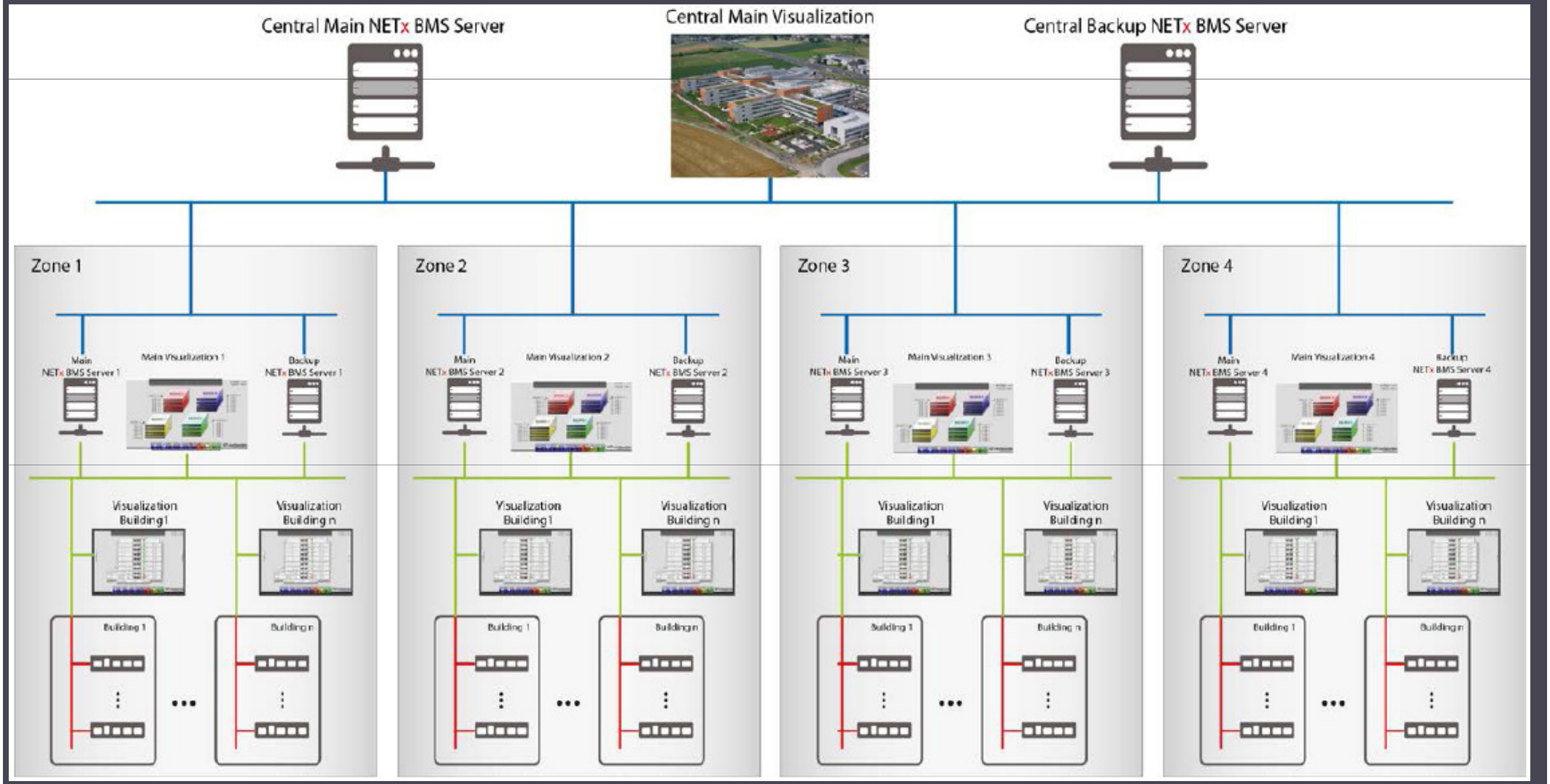
SERVICIOS TELEMÁTICOS AVANZADOS

4- EJEMPLO BMS MULTIPROTOCOLO



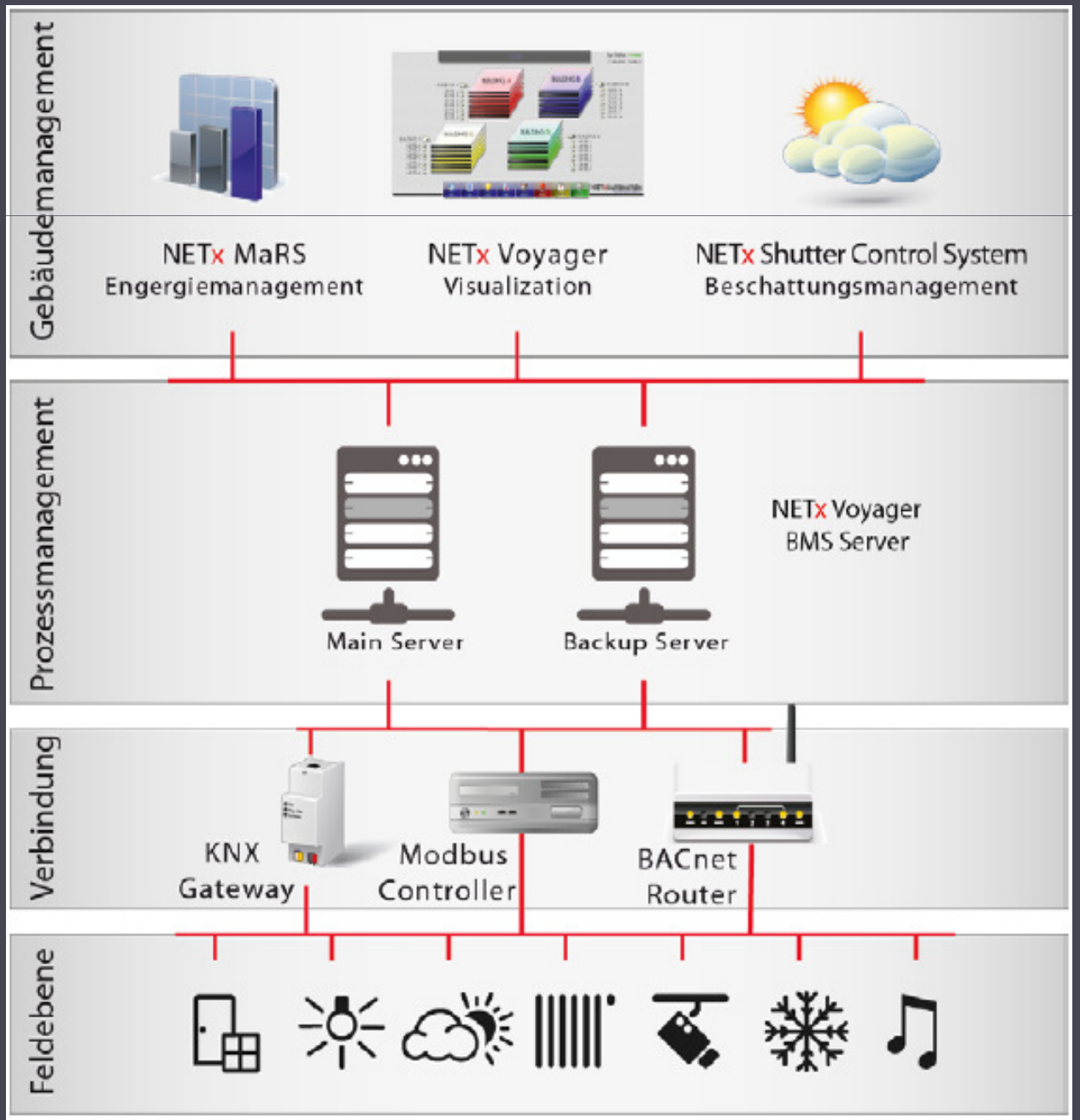
SERVICIOS TELEMÁTICOS AVANZADOS

4- EJEMPLO BMS MULTIPROTOCOLO



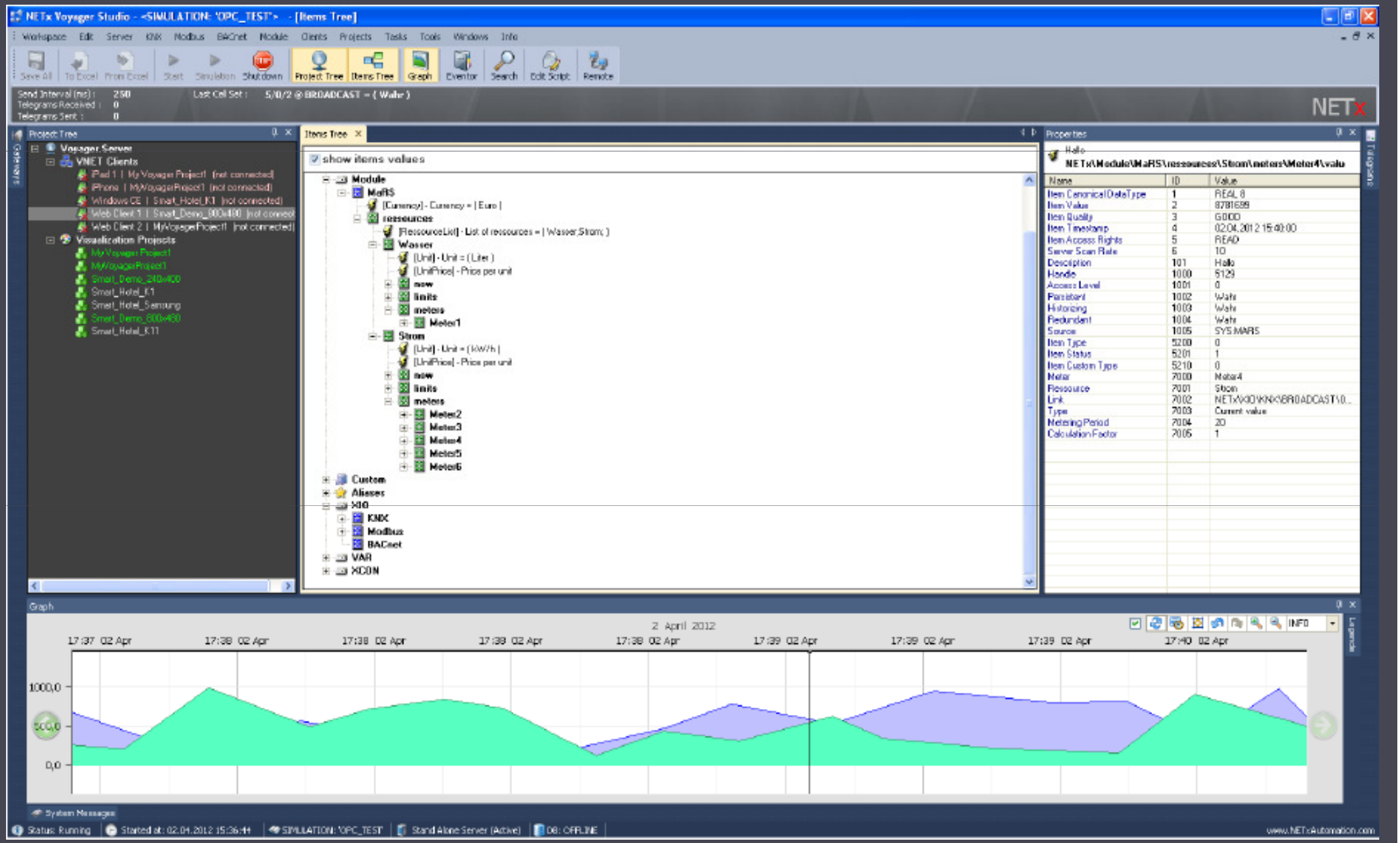
SERVICIOS TELEMÁTICOS AVANZADOS

4- EJEMPLO BMS MULTIPROTOCOLO



SERVICIOS TELEMÁTICOS AVANZADOS

4- EJEMPLO BMS MULTIPROTOCOLO



SERVICIOS TELEMÁTICOS AVANZADOS

4- EJEMPLO BMS MULTIPROTOCOLO